Microprocessor Based Interactive Process Identification Package

by

Mohammed Riyad Al-Hafeez

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

SYSTEMS ENGINEERING

May, 1985

Order Number 1355721

Microcomputer based interactive process identification package

Al-Hafeez, Mohammed Riyad, M.S.

King Fahd University of Petroleum and Minerals (Saudi Arabia), 1985

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

" بسم الله الرحمن الرحيم "

University of Petroleum and Minerals

MICROCOMPUTER BASED INTERACTIVE PROCESS IDENTIFICATION PACKAGE

BY

MOHAMMED RIYAD AL-HAFEEZ

A Thesis Presented to the
FACULTY OF THE COLLEGE OF GRADUATE STUDIES

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE IN

SYSTEMS ENGINEERING

MAY 1985

UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN, SAUDI ARABIA

COLLEGE OF GRADUATE STUDIES

This thesis, written by

MOHAMMED RIYAD AL-HAFEEZ

under the direction of his Thesis Committee, and approved by all its members, has been presented to and accepted by the Dean, College of Graduate Studies, in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
IN
SYSTEMS ENGINEERING

جامعة البترول والمعادن
كلية الدراسات العليا
COLLEGE OF GRADUATE STUDIES
UNIVERSITY OF PETROLEUM & MINERALS

Dean, College of Graduate Studies

Date: May 18th 1985.

Department Chairman

Thesis Committee

Chairman

Member

Member

Dedicated to

My Loving Father & Mother.

ACKNOWLEDGEMENTS

I am grateful to all the people who assisted and helped me during the course of my Master's thesis work and I thank them all.

The informative discussion and valuable suggestions given by Mr.Chris Warnock and other ARAMCO process engineers during the initial stages of the project are greatly appreciated.

Special thanks to Dr. Norman E. Gough, who served as my thesis committee Chairman, and assisted me with his useful suggestions and ideas. I sincerely appreciate the help given to me by Dr. M.Shaghir Ahmed in the identification and estimation part of the thesis. I also thank Dr. S.Abdul Hay for serving in my thesis committee. Thanks are due to Dr. Ala H.Al-Rabeh, Chairman, Systems Engineering Department for his generous support in assigning equipment and facilities for this research.

Lastly, I extend my deepest gratitude to my family and close family friends whose constant interest and moral support was a great source of encouragement.

ABSTRACT

A systems approach for solving problems of process control engineers in process diagnostics, graphics and computational power is applied here. A popular microcomputer is used to develop an interactive but powerful process identification package.

Stochastic time series analysis methods are employed for identification and prediction of processes. Box and Jenkins technique is implemented as it is interactive and requires little knowledge about the process itself. Adequate and parsimonious transfer function models obtained from this method are used in forecasting using leading indicators. As an alternative to Box and Jenkins the automatic technique of Pandit and Wu is implemented. From Pandit and Wu's procedure transfer function, open/close loop relationships and dual-input single-output models can be obtained. Both techniques are tested using real data.

Identification routines are developed in FORTRAN, graphics, communication and integration is achieved using multi-purpose Symphony integrated package. The whole system is developed in menu driven and user friendly environment for immediate use in process industries.

ملخـــــص

طريقه منهجيـه لحل مشاكل مهندسي تحكم العمليـات في التشخيـــص بواسطة استخدام الرسومات والقدره الحسابيه للحاسبات الآليه . يستخـــدم حاسب آلي صغير واسع الانتشار لتطوير طريقة تحادثيةوقويـه لتحديد نـــوع العمليـــــه .

تستخدم المتسلسلات الزمنيـــه لتحديد النوع والتنبؤ . وبالـــذات فان طريقة بوكس وجنكنز مستخدمه لانها تحادثيه وتتطلب قدرا أقل مـــــن المعلومات . تعطى هذه الطريقه نماذج مناسبه وبعدد قليل من المعامـــلات لتستخدم في التنبؤ . هناك طريقه أخرى ذاتيه تستخدمه وتعرف بطريقـــــة فو وباندیت . من هذه الطريقة تشتق نماذج وعلاقات في حالة فتح واغـــلاق دائرة التحكم وللعمليات ذات مدخلين ومخرج واحد . كلا الطريقتين تستخدمـــان مع معلومات حقيقيـــــة .

البرامج مكتوبـه بلغـة فورتران . أما الرسومات والاتصالات فتقوم بهـــــا " سيمفوني " . كل الأعمال تدار باختيار الأوامر المناسبه من قائمة بطريقـة وديه للاستخدام المباشر في الصناعـــه .

TABLE OF CONTENTS

APPENDIX

LIST OF TABLES

LIST OF FIGURES

Chapter 1

BACKGROUND AND OBJECTIVES

1.1 Project Motivation

Present day process and petrochemical industries are experiencing a period of intensified competition, stringent quality and safety requirements and shrinking profit margins. The manufacturers are thus compelled to look for means to improve process productivity and efficiency with the eventual aim of higher profitability. The answer to these rising challenges and mounting tasks in the process industries lies in keeping abreast with and in implementing the latest innovations in mathematics, electronics and computing. But as with the development of other technological innovations, applications of these sophisticated new tools lags behind their theoretical formulation and verification. A survey of local petrochemical industries reveals similar prevailing trends and directions.

The control room is the indispensable and the most vital part of any process plant. It is here that important decisions regarding overall plant operation and regulation are made and implemented. Any moves to pinpoint the deficiencies or suggested improvements of the overall plant

would necessarily start from here.

In the control room a person who strongly influences the day to day operation of the plant is the process control engineer. Availability of good analytical and diagnostic tools greatly enhances the control engineer's ability to predict potential troubles and be ready with possible remedies. A detailed understanding of the manufacturing process is also a prerequisite to the task of improving plant operation through computer control. In particular, control engineers must accurately quantify cause and effect relationships between plant process variables. Design of any control strategy presupposes specific knowledge of both the steady-state and time-dependent causality between plant variables. Tuning of individual feedback and feed-forward loops degenerates into trial and error methodology unless the process has been characterized in terms of gains, dead times and time constants [1].

These important characteristics of a process can only be obtained if one has access to powerful mathematical techniques for analysis of plant historical data. A study of any variable from the process, shows that in general, it is dependent on its previous values and values of other related variables at an earlier time. In other words, the behaviour of a plant can be described by time series data. The analysis of time series, is one of the most well developed

and sophisticated analytical methods for data analysis. Researchers have successfully applied time series techniques in the forecasting and control of different processes in economics, social sciences and industry. Surveying possibilities of implementing these set of useful techniques, it appears that preconceived ideas of robustness, complexity and computational intensiveness about time series methods have hampered their sound application in the process industries.

Another key element in the control room is the process control computer. Modern process computers vary in size and behaviour and they perform a multitude of operations such as performing control, logging operating records, alarming of variables etc. Although these machines are now an indispensible part of any control room and excel in their primary purpose of control and regulation, there are still shortcomings in their potential application.

As an example of these shortcomings consider for instance the graphic requirements of a control room. Engineering graphic displays give greater insight into detailed process knowledge. Graphic packages should allow high speed, high resolution multivariable plotting with user specified scaling. Graphic software should also permit the calculation and plotting of regression lines, as well as flexibility in labeling axes. Finally, CRT screen graphic images must be

easily reproduced on hard copy in order to document the results of plant tests etc [1]. Some of these requirements are available on large process control computers, but these packages are not flexible enough and are very expensive. On smaller process computer systems graphic facilities are woefully inadequate. While many systems keep a rolling short history of selected plant variables, their trend displays are single variable, low resolution plots with fixed scaling. Control engineers find it more difficult to diagnose problems now that CRT plots have replaced chart recorders to a large extent.

As a second example, consider the problem of trying to develop application routines on a process computer. The memory size of small process computers is often too limited to compile large programs. Program coding on larger systems is laborious. Large programs must be broken into segments, program compilation and execution takes lowest system priority and the reliability and dependability of the software is so low that often small programming errors lead to system crashes.

These are examples of a number of problems, all of which pertain to limiting the goals of productivity and profitability set by manufacturers of petrochemical products. It should be stressed that in large process plants, even minor increases in efficiency can influence annual

profitability by a vast amount. It follows that an inexpensive method to overcome the difficulties and provide the control engineer with a full set of analytical tools is an urgent requirement.

It is apparent to the systems engineer that much progress can be made using the present day powerful and inexpensive microcomputers. Taking memory size as an example, it is paradoxical that a process control computer (such as the FOX-3) requiring a sizeable investment holds 64K of memory, while a home microcomputer (such as the COMMODORE-64) may have the same amount of memory at a small fraction of the cost. In fact, microcomputers have emerged which can address well upto 3M RAM memory [2] and stand side by side many mainframes. They are now sufficiently intelligent and flexible to emulate any kind of terminal or communicate with any other computer or process.

Microcomputers also offer sophisticated software in spreadsheets, database, graphics and programming languages. The ability to transfer files from a mainframe means that a microcomputer provides access to a vast software resource. Problems in computation speed are gradually being resolved by using innovative software and hardware techniques.

Before outlining the objectives for this work, the current state of the time series analysis techniques coupled with

their available software is studied.

1.2 Literature Survey

Any discussion on time series analysis raises the fundamental question "Why are time series data different from other types of data?" The main difference is that for time series data, the ordering of the data is important. The value of a variable at time t is likely to depend on the value of the variable at some earlier time, s. This dependence over time is a mixed blessing. On the one hand, standard methods requiring independently, identically distributed observations cannot be used directly; on the other hand the dependence over time can be exploited for several useful purposes including prediction, control and forecasting.

From the earliest times man has kept track of and judged the importance of passage of time but the study of time series as a science in itself is a comparatively recent development. Recording events according to a horizontal axis in which equal intervals correspond to equal spaces of time must have occured a thousand years ago. Fig.1.1 [3] is earliest diagram known in which essential concepts of a time graph are apparent. However, it was only in last century

that modern time charts like histograms were first published, Fig.1.2. [3]

In the nineteenth century the subject of theoretical statistics was still in its infancy and in a state of disarray. The influence of physics among mathematician and social scientists was so strong that they followed a deterministic approach. A phenomenon tracked through time was imagined as behaving completely under deterministic laws, and any imperfections or any failure of the theory to correspond with fact was either dealt with by modifying the theory in a deterministic direction, or was attributed to 'error' analogous to errors of observation [3]. The failure of the deterministic models to account for economic, behavioral and other processes did not put a stop to this physical approach.

It was in 1927 that Yule [4] laid the basis for modern time series analysis. Working on sunspot numbers, which obviously fluctuate in a manner that cannot be entirely due to chance, Yule was struck by the fact that the amplitudes of his series and the distances between successive peaks and waves were irregular Fig.1.3. To predict future sunspot number, instead of having a behavior in which any difference between theory and observation is attributable to transient error, he suggested that the time series be corrupted with a series of shocks which are incorporated into the future

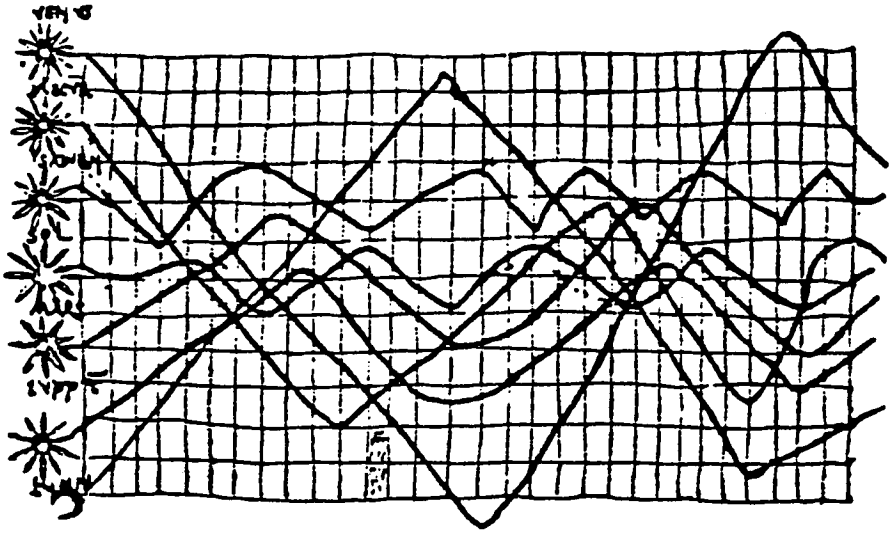


Fig. 1.1 Graph of an early time-series

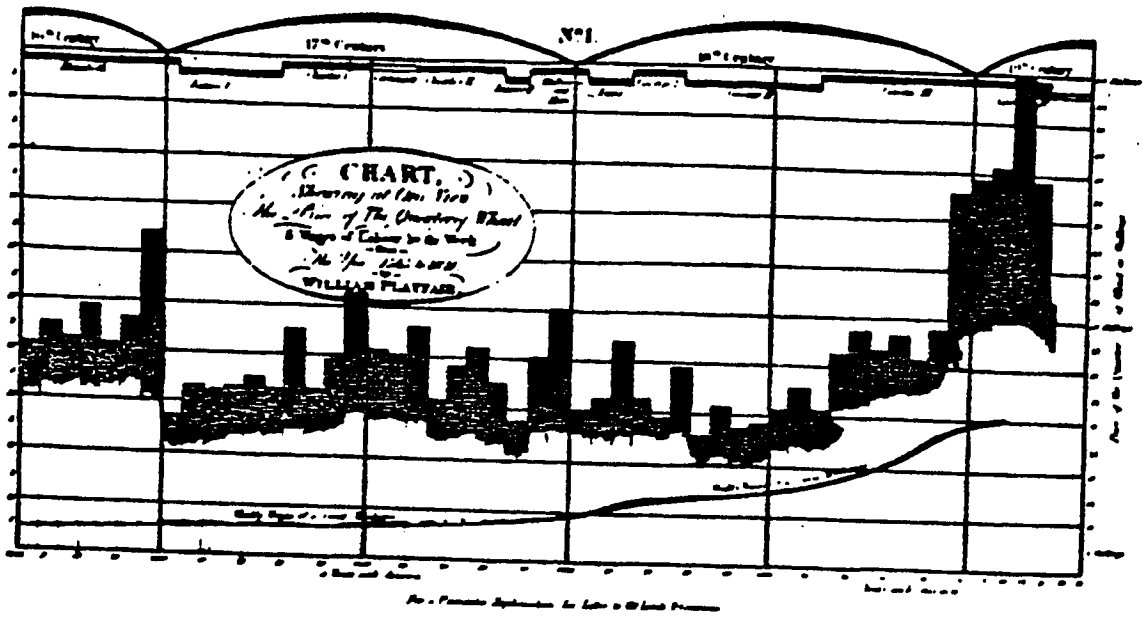


Fig. 1.2 Chart combining a graph and a histogram : from Playfair's *A Letter on our Agricultural Distress* (1821)

behaviour of the system. This concept leads to the theory of stochastic processes of which the theory of stochastic time series is an important part. Its usefulness will be exemplified many times in the sequel.

The position and importance which the time series methodology holds at present among different scientific disciplines was made possible by the work of many great mathematicians and statisticians in the last fifty years. Most of the theories developed are still in use in their original form. Around 1933 Kolmogorov presented the theory of Markov processes which laid the theoretical foundation for stochastic processes and which forms the special case of general time series models. The theory of stationary stochastic processes was advanced and developed by Khiatchin (1934). It was Wold (1938) who established a link between the statistical theory of time series initiated by Yule and the probalistic theory of the structure of time series initiated by Kolmogorov and Khiatchin. The theory put forwarded by Wold and now known as Wold's decomposition showed that an arbitrary time series can be decomposed into two parts, one (linearly) deterministic and the other stochastic, and that the stochastic part can be further decomposed as a finite or infinite moving average. Wiener's work during the 1940's carried important developments in prediction theory using the spectral representation. An account of Kolmogorov-Wiener

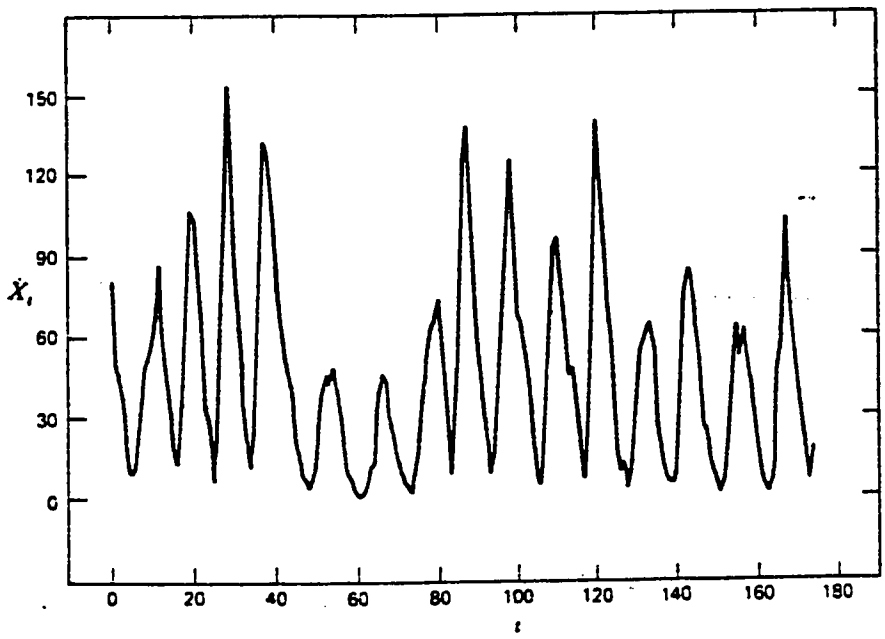


Fig.1.3 Wolfer's sunspot numbers: yearly (1749–1924).

prediction theory, not involving the sophisticated mathematics may be found in Yaglom [1952]. More recently the theory has been extended by Kalman [1960] to non-linear processes by emphasizing conditional expectation and recursive algorithms [4].

It is appropriate at this point to outline the various reasons for studying time series, as these reasons can influence the choice of the methods for a particular application. Three levels of study may be considered [3].

(a) At a superficial level, take one particular series and construct a simple system, which describes its behaviour in a concise way. Later this is termed univariate time series modeling.

(b) Penetrating a little deeper, try to develop a slightly more complex model in which a time series behaviour is explained in terms of other variables. If other variables do not fully explain the output time series, it is necessary to add a noise model which describes the discrepancies. This kind of model will be known as the transfer function.

(c) From either (a) or (b), use the resultant analysis to forecast the behaviour of the series in the future. From (a), it is assumed that, even when the basic mechanism which is generating the series is unknown, there is

sufficient momentum in the system to ensure that future behaviour will be like the past. From (b) more insight is gained into the underlying causation enabling projections to be made into the future with more confidence.

A stochastic time series can be represented in two forms statistical or spectral. For a statistical time series, the future values can be described only in terms of a probability distribution. The probabilistic laws define the mean variance and autocovariance (where autocovariance is the covariance between observation x_t and another observation separated by k units of time x_{t-k}) of the statistical time series. Spectral analysis, on the other hand, is based on the assumption that a time series is made up of sine and cosine waves with different frequencies. Spectral analysis has important aids like the periodogram, spectrum and spectral density function which is equivalent to an autocorrelation function (a normalised form of autocovariance). Spectral techniques use Fourier transform and complex arithmetic in their formulation and are thus considered more difficult to apply in practice [5]. Moreover, most time series are characterized by random changes of frequency, amplitude and phase. For this type of series, the spectral aids fluctuate wildly and are not capable of any meaningful interpretation. Also working with spectral analysis often gives models prodigal with

parameters. The time series which appear in this study are based on chemical and plant data and it has been found most convenient to describe them in terms of the autocorrelation function of the time series.

An important distinction regarding time series techniques must be made between discrete and continuous models. Throughout the coming chapters discrete models of stochastic systems are considered as they can be readily implemented on a digital computer. Discrete models can be readily obtained from a discrete set of data by extending linear methods. Moreover prediction and forecasting can be easily carried out using these models.

The publication of the book "Time Series Analysis: Forecasting and Control" by G.E.P. Box and G.M.Jenkins in 1970 [6], was a milestone in stochastic time series analysis. The authors may be credited with integrating the elements of the last half century into a comprehensive theory, extending it greatly and popularizing it [7]. Ever since this publication a vast amount of interest and effort has been exerted into time series analysis leading to extensions, improvements and applications. But up to this present day the Box-Jenkins approach is the standard against which all other works and research are measured. Earlier, three areas for time series study were outlined and each is now examined with reference to the Box-Jenkins method, to point out the various

improvements and alternatives available.

The objectives of the first part of the study are stated in [8] as:

> "Given an observed time series $[X_t : t = 1,2, \ldots N]$ and given that t satisfies a general Auto Regressive Moving Average process of order (n, m), which is abbreviate to ARMA (n,m), it is required to determine what values of n and m give the best representation of the time series $[X_t]$.

Box and Jenkins proposed efficient methods for the determination of a "pure" auto regressive (AR) process and a "pure" moving average (MA) process. Their method includes the observation of autocorrelation and partial auto correlation plots for the order determination. However for the mixed autoregressive moving average (ARMA) case they did not made precise suggestions but proposed a interactive technique which helps the user to find the correct model order. On the other hand, the empirical nature of the Box-Jenkins approach, which requires graphical and visual inspection of the time series, may mean that it may be very difficult to program their method in some situations.

Gray, Kelley and McIntire [9] have proposed an alternative approach for order determination. Their method is based on so-called R and S arrays and the G-spectral estimation.

However studies have shown that this method is quite complicated and may produce erroneous results [10].

Recently Beguin, Gourierouse & Monfort [9] have suggested the so called "Corner Method" for the order determination process. Their work proposes a new criterion based on auto correlations of the data and the building of array from auto correlations. This method seems to be simple and straight forward and works well for large sample sizes, but for moderately smaller sizes the results of the method can, at times be very misleading. This method has not been recommended for practical applications [8].

Pandit and Wu [4] have proposed powerful techniques in time series modeling. According to them the problem of order determination can be simplified and made automatic by following an ARMA (n, n-1) approach. Each series modeled is fitted with an order of increasing value of n=1,2..... The model adequacy is decided by making statistical tests among different models. A drawback of this technique is that it is computationally intensive.

The second part of this study involves the modeling of transfer functions. The main objective is to apply time series methods to process control, where accurate transfer function models are required for improving the design and control of process plants. These transfer functions regulate

important plant variables which are manipulated by on-line process control computers. Different techniques are available for transfer function identification. Stochastic (statistical) techniques for transfer function modeling are increasingly applied in the industry. These techniques are parsimonious, accurate and are easy to implement on process computers.

According to Phadke & Wu [12]

> "To achieve an optimum computer control of a system we need to identify (1) the transfer function relating each of the manipulatable variables (input variables) to the output variable and (2) the disturbances in the system which have a conglomerate effect on the variables in the system which affect the system but are not classified as manipulatable variables".

Classical methods for estimating transfer functions have been based on the choice of special inputs to the system for example, step, sine wave and pulse inputs. It has been noted [6] that these methods are useful when a system is affected by small amounts of noise, but are less satisfactory otherwise. In the presence of appreciable noise, as is the case with chemical process data, it is necessary to use statistical methods for estimating the transfer function.

From the point of view of understanding the dynamics as well as prediction and control, the modeling of processes using statistical methods and discrete operating data has the following advantages [4].

1. The normal operation of the system need not be disturbed just for the purpose of analysis.

2. The model is obtained under realistic conditions and therefore provides more precise prediction and control.

3. By continuously monitoring the data, it is possible to detect changes in the system over time and to take appropriate corrective action when necessary.

The Box-Jenkins (B-J) technique for transfer function identification consists of iterative cycles of identification, fitting and diagnostic checking to arrive at the appropriate transfer functions for input, output and the noise. A B-J model possesses simplicity in that it has a small number of parameters (parsimonious) consonant with representational adequacy . Their procedure involves studying the sample cross correlation function between the prewhitened input series and the corresponding filtered output series (filtered by the ARMA model of the input series). However, while their method is promising for the univariate case, it is difficult to generalize it for multiple input models.

Another technique referred to by Lui and Hanssen [13] as the double prewhitening method and studied by Priestley [14] and, Haugh and Box [15] consists of prewhitening both input and output series and obtaining the transfer function weights of the prewhitened series using least squares estimation. Fask and Robinson [16] and Granger and Newbold [17] have generalized this method, but inspite of its wide usage, the double prewhitening method is more difficult to apply than the Box and Jenkins method. The difficulties are that a model is frequently over-structured due to the prewhitening factors, and the fact that the lag structure is often cumbersome to derive.

Liu & Hansens [13] have described a procedure for transfer function specification for multiple-input single-output systems. Their method is based on least squares estimates of transfer function weights using the original or filtered systems. They use the corner method [11] to identify a parsimonious rational form of the transfer function. Simplicity and superior performance of the technique over the Box and Jenkins and double prewhitening methods has been claimed by Lui and Hanssens. Nevertheless, serious reservations about the corner method itself were raised by Gooijer and Hauts [8].

The so-called Data Dependent Systems (DDS) technique formulated by Pandit in his Ph.D. thesis and later advanced

by Pandit and Wu in [18], combines batch and automatic procedures for transfer function modeling. They suggest forming an Auto Regressive Moving Average Vector (ARMAV) consisting of all inputs and outputs, along the lines of the state space approach of modern control theory. The advantage in this kind of formulation is that important elements such as feedback, dead-time and interaction between different variables is evident from the state space matrices. But these techniques require tedious computation and formulation and hence are not suitable for the present microcomputer application. Fortunately a special simplified form of these models known as Extended Auto Regressive Moving Average (EARMA) models is derived in [4]. The EARMA models are often adequate to deal with one output and one or two input systems.

The last part of the study involves using earlier parts to make predictions of future events and conditions. The act of making such predictions called forecasting will be implemented to test the usefulness and accuracy of the techniques. Although forecasts are required in many areas they are only considered here in relation to process control.

Forecasting should be an important part of a process control system. Process control requires forecasts of the behaviour of the process in the future. By monitoring key process variables and using them to predict the future

behaviour of the process, it may be possible to determine the optimal time and extent of control action. For example, a chemical processing unit may become less efficient as hours of continuous operation increase. Forecasting the performance of the unit will be useful in planning the shut down time and overhaul schedule [19].

There are many techniques available for forecasting which include regression, decomposition and exponential smoothing approaches. Forecasting carried out using stochastic models is far more superior, as these models have a sound mathematical basis. As mentioned earlier, forecasting with transfer function models has the added advantage that it is based on more information about the system and hence gives more confidence about the forecasts. According to Box and Jenkins optimal forecasts can be obtained for an output time series Y_t by using information from related series X_t, which is termed the leading indicator for Y_t.

1.3 Review of Time Series Software

In the last section the development of stochastic time series methodology was outlined but the factor which has been instrumental in advancing and popularizing time series has been the power of electronic computers, which has removed the obstacles imposed by the extensive and tedious calculations

involved in time series studies [3]. Since the principal objective in this work is to develop a suitable computer software for time series analysis, it is important to study the existing state of the art in time series software.

Statistical procedures were among the earliest computer software and the introduction of Box & Jenkins technique in 1970 stimulated much of interest in implementing this time series analysis technique on computers. It was thus fortuitous that the advent to the technique paralleled the golden age of computing. Various researchers and scientists programmed the Box-Jenkins and related time series modeling technique according to their needs and available environment. The boost in computing power led to applications first on main frames, then on minicomputers of the 60's and 70's and now on the present day powerful microcomputers.

A brief survey of the available software reveals that the time series modeling programs are organised around the identification-estimation-diagnosis sequence, but significant differences exist among programs in the formats of input specifications, the presentation of output and the algorithm used for estimation. Thus in discussing time series software, it is useful for us to distinguish between four different levels of software organisation.

1. Subroutine Libraries:

A subroutine library usually consists of a series of subroutines, each of which performs a discrete and distinct statistical function. One subroutine, for example, may compute the cross correlation function for transfer function analysis while another may plot the cross correlation function. To use a subroutine library, the user must write a "main program" which reads in the data, calls the appropriate subroutine, stores the intermediate results and prints the desired output. Subroutine libraries offer great flexibility at the cost of detailed specification and programming.[20]

Foremost amongst the subroutine libraries for time series analysis is the International Mathematical and Statistical Library (IMSL). It is a large comprehensive and reliable subroutine library written in FORTRAN and its major attraction is that it is available for most of popular computer sizes and models. Recently these subroutines have been implemented on microcomputers [21].

IMSL implements both the spectral and statistical (Box & Jenkins) techniques in time series analysis. The IMSL routines are capable of carrying out univariate ARMA modeling and transfer function modeling. They contain many useful routines for identification, estimation, diagnostic checking, forecasting and control.

Stand-Alone Programs:

This is the second category according to which we can classify the available time series software. A stand-alone statistical program usually consists of a main program and its associated subroutines. It is designed to accept input from the user in a certain format, perform the necessary calculations, and print the results. The advantage of these stand-alone systems is that they are flexible, easily transportable and available at relatively low cost. The major drawback to these programs is that specification of the input parameters is usually in fixed fields and can be tedious and confusing to the inexperienced user. Stand-alone programs usually require a large amount of memory to run because of their large modules and hence it is difficult to implement them fully on smaller machines.

One of the major efforts in developing stand alone time series has been made by Pack [20,22]. The program contains two main programs which read input parameters and call appropriate subroutines for computation and display of results. Algorithms are included for the usual identification, estimation, verification and forecasting phases. In the identification step the graphical analytical methods of B-J are replaced by statistical tests. The computerized procedure is fully automatic as it determines internally all of the required decisions necessary in the B-J procedure. The authors of the program feel that the automatic

algorithm and the way it works in practice will lead to better understanding of B-J technique.

TIMSAC-78 is another stand alone program that was developed at University of Tulsa [22]. It is a package of FORTRAN subroutines and mainline programs that implement the work of Akaike [23].

TIMES BOARD is a huge time series package developed at Texas A&M University. It contains approximately 250 subroutines and ten main programs which are all in FORTRAN. Many spectral and statistical techniques are implemented. The B-J main program uses many semi-automatic criteria to guide the various stages of the analysis [22].

Batch Systems

A batch statistical system can be thought of as a series of statistical programs, each performing a certain kind of analysis. The series of programs are linked by a common set of data manipulation capabilities and a unified, 'natural' user's input language. From the researcher's point of view statistical systems are often the easiest to use; but they are also relatively costly to acquire and maintain [20].

Perhaps the greatest proof of popularity of the B-J technique can be seen by the number of major software companies offering Box-Jenkins capabilities in their batch

statistical systems. The major academic and research statistical system include such famous names as SPSS (Statistical Package for the Social Sciences), BMDP (Biomedical Computer Program), and SAS (Statistical Analysis System) [22]. The inclusion of single time series and transfer function analysis procedures in these widely distributed and well-documented systems is expected to greatly facilitate the use of time series analysis techniques by academicians and engineers alike. Each of these packages features extensive file definition and data management capabilities combined with a keyword syntax for input specifications. In SPSS and SAS, the Box-Jenkins routines are integrated into the unified system whereas in BMDP, they comprise a separate program which shares syntax and file structures with other programs. Each of these packages supports univariate Box-Jenkins analysis. SPSS and BMDP also support transfer function and intervention analysis while SAS is developing transfer function capabilities. The emergence of powerful micros gives these major software manufacturers the incentive to bring out microcomputer versions of the batch packages. Already SPSS is available on microcomputers and others are expected to follow suit. This will give much needed portability to statistical time series techniques.

Interactive Systems:

The iterative nature of Box-Jenkins time series analysis immediately suggests the desirability of interactive data analysis. In an interactive environment, the researcher ordinarily "converses" with a statistical system by means of a computer terminal key-board. A command is entered from the key-board, the desired calculations are performed, and the results are immediately displayed and/or printed at the terminal console. The analyst can then enter a new command based on interpretation of the information just received. For example in the present context, the analyst might estimate a model, perform diagnostic checking on the residuals, and then immediately estimate a revised model based on the diagnostic information just displayed. A procedure of this sort would usually require at least two separate runs of a batch program. Interactive programs are easily learned and re-learned by first time and occasional users. Many aids, such as a graphical display and on-line help are mostly available so that user has full control over the program functions. At present, interactive statistical time series systems are not distributed as widely as batch systems, which are considered nevertheless adequate. They are required mainly by a time series analyst who has to execute identification-estimation-diagnosis in sequence, with many interpretive decisions to be made at each step. It is generally accepted that flexible, interactive software which facilitates movement within and between these three steps is

optimal. Again, development is proceeding rapidly in the field of interactive systems and such systems will probably be more readily available in the future.

Of the available interactive programs MINITAB is a prominent one. It is a flexible general interactive statistical system, written in FORTRAN for research and instruction using small data sets. It supports univariate time series analysis and can be easily adapted for different sizes and models of computers. Recently it has been successfully implemented on different microscomputer [24]. SYBIL/RUNNER is an interactive system oriented primarily towards applied managerial forecasting. It includes univariate series and transfer function capabilities in its extensive forecasting repertoire. Among the major statistical batch systems BMDP has an option for running in an interactive mode. Also, SCSS, the SPSS conversational statistical system, is a very powerful interactive system that runs on several different models of computers, but time series modeling has not been implemented yet [20].

A powerful tool for interactive system implementation is the APL language and much prominent work has been done in implementing time series techniques via APL. An APL system using Box-Jenkins methodology has been implemented by Kaltio at Helsinki University of Technology [25]. The system has functions for analysis of single time series models, transfer

function models and decomposition of time series. One unique feature of the package is that it is based on a processor which executes the tasks defined by the user with a special notation designed for time series analysis.

The work done by Prins [26] at IBM is also worth mentioning. His work is based on the Pandit-Wu technique and has been programmed in APL as well. The system includes scalar systems, transfer models, open-and closed-loop systems, as well as multivariate stochastic systems. In this interactive package reliance on prior examination of auto and cross correlation or on differencing in dealing with nonstationary processes is minimized.

Another package which is based on entirely different views than the previous one has been developed by Polhemus [27]. His intention was to demonstrate that graphics can provide an effective interface between the analyst and statistical calculations, allowing judgement and experience to play an active role in time series building. The package's interactive environment for time series analysis is centered around a statistical graphics system, modeled on Box-Jenkins methodology and utilizes the powerful graphical and interactive facilities of APL language.

1.4 Thesis Objectives

The main objective of this work is to develop a microcomputer-based process identification package. It is intended that this will be a user-friendly analytical tool for practising control engineers in process forecasting and control. The package should also serve as an invaluable teaching aid in identification and control courses.

The modeling of processes is carried out using stochastic techniques for transfer function analysis. The technique of Box and Jenkins has been utilised for transfer function modeling, because of its simplicity, interactiveness and robustness. Any deficiencies in the Box and Jenkins method can be resolved by the host of modified and new techniques now available.

Implementation of the package on a microcomputer provides much needed portability and wider applicability for time series analysis techniques. The package on a microcomputer was designed to fulfill the following criteria:

a) Efficient and fast computation using a high level language.

b) Adequate graphics for trends, comparison and analysis.

c) Interface with the process to obtain data.

d) Integration of all these facilities to create a user friendly interactive system.

Chapter 2

BOX-JENKINS TECHNIQUE

The Box and Jenkins method may be defined as a comprehensive statistical technique that has been developed to derive from time series data, models for prediction, simulation and control of stochastic, or random processes.

The interactiveness and step-by-step procedures allow for full user control and flexibility. The user may go through step-by-step model building or he may try any of his own models based upon his intuition or observation. The very fact that the technique is step-by-step means that the user is unlikely to be lost or left in confusion as his input is required in many instances.

The generalised models from the B-J technique are made with minimal assumptions. Also models derived are claimed to be parsimonious in nature, so that complicated models are expressed as simple equations containing a minimum number of parameters.

B-J models are especially well suited for time series in which the sampling is small, so that a relatively long history can be easily obtained. For this reason, they have been widely applied to series where hourly, daily or weekly

observations are of interest. For example, the time series of output characteristics of chemical processes, such as yield, purity, etc. are often of this general type.

The Box and Jenkins technique can be used to obtain optimal forecasts of future values of a single series or when two or more related time series are under study. The models can be extended to represent dynamic relationships between the series and hence to estimate transfer functions.

The transfer function modeling procedure of B-J is a two-step procedure. The first step consists of modeling a univariate time series which is the input data. The second step is to modify input-output transfer function. We begin by introducing some important background material and then discuss various type of time series models. Next we present a univariate time series modeling strategy, which is later expanded to transfer function models. Finally we use forecasting methods to observe how well the models work in predicting the future output.

2.1 Background Material

In this section basic methods and definitions needed for the analysis of time series are introduced. These basic definitions help to build the Box-Jenkins methodology for transfer function analysis.

2.1.1 Backward Shift Operator:

We shall employ extensively the backward shift operator B which is defined as:

$$BX_t = X_{t-1} \qquad (2.1.1)$$

The expression means that B operaters on X_t to shift it backward one point in time. The properties of the backward shift operator are:

$$(B^n)X_t = X_{t-n}$$

$$(B^n B^m)X_t = X_t(B^{n+m}) = X_{t-n-m}$$

The operator obeys all the laws of exponents that are routinely used in polynomial algebra. We will use the operator to difference the time series:

$$(1-B)X_t = X_t - X_t(B) = X_t - X_{t-1} = Z_t$$

Another useful property of the backward shift operator is invertibility. The property of invertibility allows movement forward and backward in time by applying the operator and its inverse to a time series. The forward shift operator is denoted by B^{-1} and it has the same properties as the backward shift operator. Also

$$B^{-1}B = 1$$

$$BB^{-1}X_t(B) = BX_{t+1} = X_t$$

In the following sections of this chapter and later chapters, the backward shift operator is used routinely to describe time series models. The operator greatly simplifies the algebra and, indeed, some of the higher order ARMA models cannot be written economically without the operator convention.

Autocovariance and Autocorrelation:

If X_t denotes a stochastic time series with zero mean, the autocovariance function $\gamma(k)$ or γ_k is defined as:

$$\gamma_k = E[X_t X_{t-k}] \quad , k = 0,1,2$$

For a stationary time series the autocorrelation function ρ_k, is

$$\rho_k = \frac{\gamma_k}{\gamma_0} \tag{2.1.2}$$

Clearly, the autocovariance function is simply the covariance between current and logged values of the time series in question, and the autocorrelation function is the autocovariance function standardized by division by γ_0. Since autocovariance function is not normally known, they must be estimated. Consistent estimates are given by:

$$C_k = \hat{\gamma}_k = \frac{1}{N} \sum_{t=k+1}^{N-k} (X_t - \bar{X})(X_{t-k} - \bar{X}) \tag{2.1.3}$$

$$\gamma_k = \hat{\rho}_k = \frac{C_k}{C_0} = \frac{\hat{\gamma}_k}{\hat{\gamma}_0}$$

Autocorrelation functions are usually plotted. Different models give rise to characteristic autocorrelation patterns and hence they may be used to recognize which model to fit to a series initially.

2.1.3 Stationarity:

Modeling and fitting of time series data using stochastic modeling techniques requires that the data be stationary with no seasonal trend. If a time series is stationary, its value fluctuates around a constant mean μ. Conversely a non-stationary time series has no constant mean. A check for stationarity can be made by plotting the data and observing the trends. If the autocorrelation plots of time series die out slowly, non-stationary behaviour is suspected. Differencing is mostly applied to remove the non-stationary behaviour. The (backward) difference operator ∇ as:

$$\nabla x_t = x_t - x_{t-1} \qquad (2.1.4)$$

or in terms of backward shift operator B as

$$\nabla = 1 - B$$

The higher-order differencing can be expressed as:

$$\nabla^2 = (1 - B)^2$$

or

$$\nabla^2 x_t = (1 - B)^2 x_t = (1 - 2B - B^2) = x_t - 2x_{t-1} - x_{t-2}$$

Differencing a time series (x_t) on length n produces a new time series $w_t = \nabla^d x_t$ of length n-d, where d is the degree of differencing.

Sometimes the time series data possess seasonal variations. Defining L to be the number of seasons in a year, then L = 12 for monthly data and L = 4 for quarterly data. The seasonal operator ∇_L is defined as

$$\nabla_L = (1 - B^L)$$

or

$$\nabla_L x_t = x_t - x_{t-L}$$

In general

$$\nabla_L^D x_t = (1 - B^L)^D x_t$$

where D is any degree of seasonal differencing that may be required to produce stationary time series values.

2.2 Time Series Models

To analyze a time series it is necessary to specify time series models. These models impose a structure on the time

series that makes it possible to using a finite number of parameters, to characterize, analyze, and forecast using a time series [20]. Four different, although related, parameterizations of time series are now developed: the autoregressive model, the moving average model, the autoregressive moving average model, and the transfer function model.

2.2.1 Autoregressive Model:

The autoregressive structure is a relatively simple, but powerful model for the analysis of a time series. The autoregressive process assumes that the current value of a time series depends upon past values of the time series and a random shock.

An autoregressive model of order n, AR(n) is expressed as

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots\ldots \phi_n x_{t-n} + a_t \qquad (2.2.1)$$

where a_t is a white noise process with zero mean and variance σ_{a^2}.

The AR(n) process can be written in terms of backward-shift operators as:

$$x_t = (\phi_1 B + \phi_2 B^2 + \ldots\ldots \phi_n B^n)\ x_t + a_t$$

or

$$(1 - \phi_1 B - \phi_2 B^2 \ldots\ldots \phi_n B^n)\, x_t = a_t$$

Denoting

$$\Phi_n(B) = 1 - \phi_1 B - \phi_2 B \ldots\ldots \phi_n B^n$$

equation (2.2.2) may be written as

$$\Phi_n(B)\, x_t = a_t \qquad (2.2.3)$$

The autoregressive model may be used to represent both stationary and non-stationary time series. Box and Jenkins show that if the roots of the polynomial $\Phi(B)=0$ lie outside the unit circle, then the process is stationary.

The autocovariance and the autocorrelation functions can be used to shed some light on the dynamic properties of the AR(n) process - writing out the AR(n) process (2.2.1) and multiplying by X_t yields:

$$X_t X_t = \phi_1 X_t X_{t-1} + \phi_2 X_2 X_{t-2} + \ldots\ldots \phi_n X_t X_{t-n} + X_t a_t$$

Upon taking expectations, and recalling that $E[X_t X_{t-k}] = \gamma_k$

$$\gamma_o = \phi_1 \gamma_1 + \phi_2 \gamma_2 + \ldots\ldots\ldots + \phi_n \gamma_n + \sigma_a^2$$

Multiplying X_t by X_{t-k}, and after taking expectation,

$$\gamma_k = \phi_1 \gamma_{k-1} + \phi_2 \gamma_{k-2} + \ldots\ldots\ldots \phi_n \gamma_{k-n} \qquad k \geq 0 \qquad (2.2.4)$$

On dividing throughout by γ_0 it is seen that the autocorrelation function satisfies the same form of difference equation

$$\rho_k = \phi_1\rho_{k-1} + \phi_2\rho_{k-2} + \cdots\cdots \phi_n\rho_{k-n} \qquad k > 0$$

The Partial Autocorrelation Function:

The Partial Autocorrelation function an AR(n) process can be described in terms of n non-zero functions of the autocorrelations denoted by ϕ_{kj}, the jth coefficient in an autoregressive process of order k, so ϕ_{kk} is the last coefficient. From (2.2.4) the ϕ_{kj} satisfy the set of equations:

$$\rho_j = \phi_{k1}\,\rho_{j-1} + \cdots\cdots + \phi_{k(k-1)}\,\rho_{j-k+1} + \phi_{kk}\rho_{j-k} \qquad j=1,2\ldots.k \tag{2.2.5}$$

The first n equations known as the Yule-Walker equations can be written in matrix form as [6]

$$\begin{bmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-2} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & 1 \end{bmatrix} \begin{bmatrix} \phi_{k1} \\ \phi_{k2} \\ \cdot \\ \cdot \\ \cdot \\ \phi_{kk} \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \cdot \\ \cdot \\ \cdot \\ \rho_k \end{bmatrix} \tag{2.2.6}$$

or

$$\rho_k \phi_k = \rho_k$$

Solving these equations for k = 1,2,3... successively, yields ϕ_{kk}. This, is called the partial autocorrelation function. For autoregressive model n, all partial autocorrelations for lag k greater the n are zero and hence they may be used to provide initial guesses of the order of the model. As with autocorrelation functions, the partial autocorrelation function plots are also indispensable in the Box-Jenkins technique.

2.2.2 Moving Average (MA) Models:

The moving average model represents a time series as a weighted sum of current and past white noise random shocks. For a moving average process a series X_t is expressed as a linear function of past a_t terms.

$$x_t = a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} \cdots\cdots \theta_q a_{t-q}$$

$$= \sum_{i=0}^{m} \theta_i a_{t-i}$$

$$= \Theta(B)\, a_t$$

where

$$\Theta(B) = 1 - \theta_1 B - \theta_2 B^2 \cdots\cdots\cdots\cdots \theta_m B^m$$

A moving average process that depends upon lagged a_t terms, where the largest lag is m, is referred to as an MA(m) process. The representation of X_t expresses the time series as a (possibly infinite) linear combination of white noise terms, where white noise process is defined as a series of independently, identically distributed random variables.

There is an interesting duality between the moving average and autoregressive processes, related by invertibility conditions. Considering the MA(1) process for example,

$$\hat{x}_t = a_t - \theta_1 a_{t-1}$$

$$= (1 - \theta_1 B)\, a_t$$

$$a_t = (1 - \theta_1 B)^{-1}\, \hat{x}_t$$

If $|\theta_1| < 1$ then

$$a_t = \left(\sum_{i=0}^{k} \theta_i B^i \right) \hat{x}_t$$

or

$$a_t = (1 + \theta_1 B + \theta_2^2 B^2 + ...)\, x_t \qquad (2.2.8)$$

which is an infinite-order autoregressive process with weight $\phi_j = \theta^j$ The MA(1) process has been inverted to obtain a AR(∞) process with the condition

$$|\theta_1| < 1$$

This is called the invertibility condition for an MA(1) process. General invertibility conditions imply that an MA(m) process is stationary regardless of the values of the weights θ_i, but is invertible only if the roots of $\theta_m(B)=0$ are outside the unit circle. Also the AR(n) process is stationary only if the roots of $\phi_n(B) = 0$ lie outside the unit circle, but is invertible for all values of the weights ϕ_i.

The autocorrelation function for the MA(m) process is in general non-zero for the first m terms, but zero for terms higher than m. The partial autocorrelation function is non-zero for all terms. Plots of autocorrelation functions and partial autocorrelation functions for several moving average processes are presented. It is noted that in all cases the autocorrelation function is zero for lags that exceed m and that in all cases the partial autocorrelation function approaches zero but that the speed and manner with which it approaches zero depends upon the particular coefficients.

2.2.3 The Auto Regressive Moving Average Process:

In building an empirical model of an actual time series, it is usually seen that inclusion of both autoregressive and moving average terms leads to a more parsimonious model than could be achieved with either the pure autoregressive or pure

moving average forms. This results in the mixed autoregressive moving average model of order (n,m):

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_n x_{t-n} + a_t \qquad (2.2.9)$$

$$- \theta_1 a_{t-1} - \theta_2 a_{t-2} \ldots\ldots - \theta_m a_{t-m}$$

or

$$\Phi_n(B)\, x_t = \Theta_m(B)\, a_t$$

which is abbreviated ARMA(n,m). The stationary and invertibility conditions for the AR(n) and MA(m) processes establish these properties for the ARMA(n,m) process. That is, ARMA(n,m) is stationary if the roots of $\Phi_n(B) = 0$ lie outside the unit circle, and invertible if the roots of $\theta_m(B) = 0$ lie outside the unit circle.

Although both the autocorrelation function and the partial autocorrelation function are of infinite length, the autocorrelation function and the partial autocorrelation function still yield information about the order of the ARMA(n,m processes). Writing out the ARMA(n,m) process and multiplying by Y_{t-k} yields:

$$X_t\, X_{t-k} = \phi_1 X_{t-1}\, X_{t-k} + \phi_2 X_{t-2} X_{t-k} + \ldots\ldots \phi_n X_{t-n} X_{t-k}$$

$$+ a_t\, X_{t-k} + \theta_1 a_{t-1} X_{t-k} + \ldots\ldots \theta_m a_{t-m}\, X_{t-k}$$

If $k > q$, the moving average parts of X_t and X_{t-k} are

independent. Taking expectations on both sides of the above equation yields:

$$\gamma_k = \phi_1\gamma_{k-1} + \phi_2\gamma_{k-2} + \cdots\cdots \phi_n\gamma_{k-n} \qquad k > m \qquad (2.2.10)$$

This is the same relationship that is obtained from an AR(A) process, so that the autocorrelation function of an ARMA(n,m) process behaves the same way as the autocorrelation function for an AR(n) process beyond the m-th term. For the first m terms the autocorrelation function of an AR(n) and ARMA(n,m) behaves differently, but beyond the first m terms the autocorrelation function of an ARMA(n,m) process behaves just like the autocorrelation function of an AR(n) process. That is for a stationary process, the autocorrelation function decays exponentially or sinsoidally as determined by the largest root of the difference equation shown above.

Auto Regressive Integrated Moving Average Models (ARIMA)

Often time series must be differenced, and then ARMA are fitted to the differenced series. So frequently is this done that a special name is given to models fitted to differenced data. An ARMA (n,m) model fitted to the d-th difference of a data series is called an autoregressive integrated moving average (ARIMA) model of order (n,d,m) written as ARIMA(n,d,m). The ARIMA(n,d,m) process is defined by

$$w_t = \phi_1 w_{t-1} + \ldots\ldots \phi_n w_{t-n} + a_t - \theta_1 a_{t-1} \ldots\ldots \theta_m a_{t-m} \quad (2.2.11)$$

with

$$w_t = \nabla^d x_t$$

For a wide variety of series, differencing is likely to make a series stationary, to be ready for modeling. In particular ARIMA(n,d,m) models are capable of describing series whose level, slope, and if necessary, higher derivatives, are being continuously modified, or adapted, by random shocks entering the system.

2.2.4 Transfer Function Models:

Transfer function models are used for relating an output time series, which is to be forecast and controlled, to a set of related input variables. Those models enable a time series to be forecast not only from its own past history but also from the past history of other related variable.

Let Y_t be an output variable and X_t be input variable which is to be related to the output. Then Y_t may be split into two components [28]:

$$Y_t = U_t + N_t$$

where

U_t contains that part of Y_t which can be explained exactly in terms of X_t

N_t is an error or noise term which cannot be explained

in terms of X_t - it represents all the 'missing' variables.

Consider first the relationship between U_t and the input X_t. A general way of representing a linear dynamic relationship of this kind is

$$U_t = \delta_1 U_{t-1} \ldots\ldots \delta_r U_{t-r} = \omega_0 X_{t-b} - \omega_1 X_{t-b-1} \ldots \omega_s X_{t-b-s}$$

that is

$$U_t = \frac{\omega_0 - \omega_1 B \ldots\ldots \omega_s B^s}{1 - {}_1B \ldots\ldots {}_rB^r} X_{t-b} \qquad = V(B)X_t \quad (2.2.13)$$

where

$$V(B) = \frac{\omega(B)}{\delta(B)} B^b$$

The transfer function U(B) contains

i) a 'moving average' operator $\omega(B)$,

ii) an 'autoregressive' operator $\delta(B)$,

iii) a pure delay parameter b, which represents the number of complete time interval before a change in X_t, begins to have an effect on Y_t.

Moreover both input and output series might posses non-stationarities so different levels of differencing will be needed to transform the data.

$$\nabla^d Y_t = \frac{\omega(B)}{\delta(B)} \nabla^d X_{t-b} + N_t$$

Generally noise will be non-stationary and hence in many practical situations could be represented by an ARIMA(n,d,m) model of the form

$$\nabla^d N_t = \frac{\theta(B)}{\phi(B)} a_t$$

Thus a transfer function noise model is obtained:

$$Y_t = \frac{\omega(B)}{\delta(B)} X_{t-b} + \frac{\theta(B)}{\phi(B)} a_t \qquad (2.2.15)$$

Cross Correlation Function

It is sometimes useful to think of autocorrelation as within-series correlation. In the same way that the autocorrelation function is used to identify within-series correlation, the cross correlation function is used to identify between-series correlation. Patterns of between-series correlation are used to identify a transfer function relationship between two time series in much the same way that the autocorrelation function is used to identify an ARIMA relationship within the time series.

If two time series X_t and Y_t are stationary, the cross correlation coefficient γ_{xy} measures the correlation between X_t and Y_{t+k} or in other words it measures the extent to which

a value of the first series above or below the mean tends to be followed by a value of the second series above or below the mean K time units later. An estimate of the cross correlation function can be found by the expression

$$r_{xy}(k) = \frac{c_{xy}(k)}{s_x \, s_y} \qquad K = 0, \pm 1, \pm 2 \ldots K \qquad (2.2.16)$$

where $s_x = \sqrt{c_{xx}(0)}$ and $s_y = \sqrt{c_{xx}(0)}$ are the variance of series x_t and y_t and $c_{xy}(k)$ cross covariance between two series defined by:

$$c_{xy}(k) = \begin{cases} \frac{1}{N} \sum_{t=1}^{N-K} (x_t - \bar{x})(y_{t+k} - \bar{y}) & K = 0,1,2 \\ \\ \frac{1}{N} \sum_{t=1}^{N+K} (y_t - \bar{y})(x_{t-k} - \bar{x}) & K = 0,-1,-2 \end{cases} \qquad (2.2.17)$$

The cross correlation plots provide vital information in helping to identify autoregressive order r and moving average order s. The dead time or time delay 'b' is also observed from the cross correlation plots. Box and Jenkins have given theoretical plots for different values of r and s, so that by comparison between the theoretical and estimated plots it is possible to identify the tentative order of the transfer function model.

2.3 Modeling

After discussing the basic concepts and various time series models it is now appropriate to develop the B-J transfer function modeling strategy. This strategy can be essentially sub-divided into two different but equivalent parts of univariate ARIMA time series modeling and transfer function modeling using the concepts of pre-whitening and cross correlation function.

Each of these approaches consists of a three-stage iterative procedure. First a tentative model of the ARIMA class is identified through analysis of historical data. Then unknown parameters of the model are estimated. Finally, diagnostic checks are performed to determine the adequacy of the model, or to indicate potential improvements. Each of these steps for the two parts of transfer function modeling are now discussed in more detail.

2.3.1 Univariate ARIMA Modeling

(i) Identification

Up to now the discussion concerned the formulation of ARIMA (n,d,m) models. To use the ARIMA(n,d,m) model it is necessary to work with estimated parameters, but before an ARIMA(p,d,q) model can be estimated it is essential to know the order of the process, that is n,d and m. It is of course, possible in principle to search over all possible models to see which process fits best, but even for restricted values

of n,d and m the permutations of models to fit increases very rapidly. To overcome this problem, Box and Jenkins developed a set of techniques known as identification techniques.

These techniques are the key to model building and an ARIMA model must have some empirical basis i.e. put simply, there should be some reason for selecting one tentative model over another. The empirical basis will ordinarily be the patterns of autocorrelation and partial autocorrelation functions estimated from the time series. To identify a tentative ARIMA(n,d,m) time series model satisfactorily here, actual historical data is used with at least 50 sets of data.

Once the sample autocorrelation and partial correlation function have been computed from equations (2.1.3)(2.2.6) derived in the last section, they are displayed graphically and a tentative model is identified by comparing the observed pattern with the theoretical autocorrelation and partial autocorrelation patterns. These theoretical patterns [19] are shown in Table 2.1.

Table 2.1- Behaviour of theoretical autocorrelation and partial autocorrelation functions for stationary models.

Model	Autocorrelation Function	Partial autocorrelation Function
AR(n)	Tails off	Cuts off after lag n
MA(m)	Cuts off after lag m	Tails off
ARMA(n,m)	Tails off	Tails off

The expression 'tails off' means that the function decays in an exponential, sinusoidal, or geometric fashion, approximately, with a relatively large number of non zero values. Conversely, 'cuts off' implies that the function truncates abruptly with only a few non-zero values. The duality between autoregressive and moving average processes is apparent in the above table.

If the time series is non-stationary, the sample autocorrelation function will die down extremely slowly and will not exhibit the pattern mentioned above. This is because in any realization of a non-stationary series the observations will tend to be on the same side of the sample mean for many periods, and, as a result, large sample autocorrelations at very long lags are produced. The remedy for this situation is to take the first difference of the series and compute the auto-and partial autocorrelation functions. If these function display the theoretical behaviour of Table 9-1, then one difference is necessary to produce stationarity. If not, it is necessary to successively try higher orders of differencing until stationary behaviour is achieved.

The standard errors(S.E) of the sample autocorrelation and partial autocorrelation functions are useful in identifying non zero values. As a general rule, it will be assumed that an autocorrelation or partial autocorrelation coefficient is

zero if the absolute value of its estimate is less than

$$S.E. = \pm 2/\sqrt{N}$$

where N is the number of data points. Plotting the limits directly on the graph of these functions, helps to identify non-zero values.

It has been generally noted that the identification of ARIMA(n,d,m) from autocorrelation and partial autocorrelation plots is ambiguous, and difference in opinion may emerge between two analysts after seeing the same plots. But much of this confusion can be removed in later stages of model building and moreover as pointed out in [5], identification of the appropriate ARIMA model requires skill obtained by experience.

(ii) Estimation

In this section, the estimation of previously identified ARIMA models is discussed.

Estimation follows identification. Having tentatively identified ARIMA(n,d,m) model for the time series, the ϕ and θ parameters of the tentative model must be estimated. An ARIMA(n,d,m) model is linear in parameters, but the shocks must also be estimated and hence a non-linear algorithm is

required. Any non-linear algorithm requires initial estimates of the model parameters.

(a) The Preliminary Estimates:

The preliminary estimates of parameters are obtained through the relationships that link model parameters and autocorrelations. The procedure consists of replacing the autocorrelation function by its estimate and solving for the unknown model parameters. The preliminary estimates speed up convergence and minimize the chance that convergence takes place towards a local minimum in the estimation routine. In addition preliminary parameter estimates also give the analyst an idea of how the final model will look.

Essentially for an ARMA(n,m) process preliminary estimates of ϕ and θ are sought. Box and Jenkins have proposed finding the preliminary estimates for the autoregressive part first and then for the moving average part.

For a stationary series x_t, with its autocovariance c_t defined for k = 0,1 ..., K where K $\geq$ n + m. Then if n>0, solving the set of n linear equation

$$A\ \phi_0 = x \qquad (2.3.2)$$

where

$$A_{ij} = c_{|m+i-j|}$$

$$x_i = c_{m+i}$$

$$i,j = 1,2,.....n$$

gives preliminary estimates for the autoregressive parameters. The estimates of moving average parameters are computed from the roots of the non-linear system.

$$f_i(\theta_0,.....,\theta_m) = \sum_{j=0}^{m-i} \theta_i\theta_{i-j} - c_i' \qquad i = 0,1....m \tag{2.3.3}$$

The values c_t' are modified covariances which are computed by the formula

$$c_i = \begin{cases} \sum_{i=0}^{n} \sum_{k=0}^{n} \phi_i\phi_k c_{|i+j-k|} & \text{for } n > 0 \\ c_{i+1} & \text{for } n = 0 \end{cases}$$

One should exercise caution in drawing inferences from such models, however, as the final non-linear least squares estimates of the model parameters may differ considerably from the preliminary estimates.

(b) Non-Linear Estimates:

The initial estimates are found from the above section for the ARIMA model (2.2.11)

$$x_t = \sum_{i=1}^{n} \phi_i x_{t-i} + a_t - \sum_{i=1}^{m} \theta_i a_{t-i}$$

where a_t are the residuals or one-step forecasting errors. A modified Gauss Newton algorithm [29][Appendix A]then finds the ϕ and θ in the above model which minimize:

$$S(\phi,\theta) = \sum_{t=0}^{N} [a_t|\phi,\theta,x]^2 = \sum_{t=0}^{N} [a_t]^2 \qquad (2.3.5)$$

Convergence is assumed if the change per iteration of the parameters become negligible.

The parameter estimates from the above routine should be statistically significant, i.e. any parameter whose estimated value is not significantly different than zero should be dropped from the tentative model. Also the estimated ARIMA model must lie within the bounds of stationarity and invertibility. If the estimated parameters of the tentative model do not satisfy the stationarity-invertibility conditions, then the tentative model must be rejected. The question of rejection or acceptance of the model is tackled in the next section of diagnostic checking.

(*iii*) *Diagnostic Checking*:

After a tentative ARIMA model has been fitted to the univariate time series, it is important to perform diagnostic checks to test the adequacy of the model and, if need be, to

suggest potential improvements.

If the fitted model is adequate, it should transform the observations to a white noise process. Thus a logical test is to analyze the residuals, i.e. the differences between the observed data and the predictions given by the tentative model. The series $a_t = x_t - \hat{x}_t$ is calculated from the non-linear estimation routine and the autocorrelation function plots is examined.

Let $\rho_{\hat{a}\hat{a}}$ be the sample autocorrelation function of the residuals. If the model is adequate, then the residual sample autocorrelation function should have no structure to identify, that is the autocorrelation should not differ significantly from zero for all lags greater than one. If the form of the model were adequate and the true parameter values were known, then the standard error of the residual autocorrelations would be $\pm 2/\sqrt{N}$ given N is the number of data points. If any of the residual autocorrelation are non-zero, the apparent structure would be incorporated into the original model and the series refitted. This process is continued until the residual autocorrelations resemble those of a white noise process.

Specifically, it has been found that an effective way to measure the overall adequacy of the tentative model is to examine a quantity that determines whether the first K

autocorrelations of the residuals, considered together, indicate adequacy of the model. This quantity is the well-known Box-Pierce Chi-Square Statistic, denoted by the symbol Q, and computed using the formula [6]

$$Q = (N - d) \sum_{k=0}^{k} \rho^2_{\hat{a}\hat{a}}(k) \qquad (2.3.6)$$

where N -is the number of observations in the original time series,

d -is the degree of differencing used to transform the original time series into a stationary time series,

$\rho_{\hat{a}\hat{a}}(k)$ -is the sequence of autocorrelation of the residuals at lag k.

A small value of Q indicates small autocorrelation and thus adequacy of the model. The larger the value of Q, the larger are the autocorrelations between the residuals and the more related are residuals. Thus a large value of Q indicates that the model is inadequate. It is common practice to accept the adequacy of the model if the calculated value of Q is less than $\chi^2_{0.05(K-n-m)}$ which is defined to be the point on the scale of the chi-square distribution having K-n-m, degrees of freedom such that there is an area of 0.05 under the curve of this distribution above this point. Here, np = n+m is the number of parameters that must be estimated in the model under consideration. The choice of K, the number of residual autocorrelations used in the calculation of Q, is

arbitrary, but common practice is to compute Q for K = 15 (or possibly 30).

2.3.2 Transfer Function Modeling:

For univariate ARIMA modeling, the Q statistic test in the last section is the final step in the modelling procedure. However the eventual aim is to obtain a transfer function and hence this section builds upon the techniques of the last section, to arrive at transfer function modeling procedure.

Assuming that a satisfactory model for the input series x_t, has been obtained from the methods described in the last section, attention is next focussed on the output series y_t. The first step is to make the output series y_t stationary. The usual procedure is to examine a plot of the data and then observe the auto-and partial autocorrelation plots of the output series. If the autocorrelations die out slowly, differencing is carried out until stationarity is achieved.

(i) Prewhitening and Identification:

Once stationarity of the input and output series has been achieved cross correlation is used to identify the transfer function. However the cross correlation function between two correlated time series often contains spurious values and hence can provide a misleading picture of the relationship between the two time series. To avoid such spurious cross

correlations the univariate stochastic model for the input series is used to convert x_t into a random series α_t, i.e.is to prewhiten the input. The same input model applied to the output series will convert it to a series β_t. The α_t and β_t series may be found by the formulae

$$\alpha_t = x_t - \sum_{i=0}^{n} \phi_i x_{t-i} + \sum_{j=1}^{m} \phi_j \alpha_{t-j} \qquad (2.3.7)$$

$$\beta_t = y_t - \sum_{i=1}^{n} \phi_i y_{t-i} + \sum_{i=1}^{m} \phi_j \beta_{t-j}$$

The prewhitened cross correlation function is the cross correlation function between the prewhitened input x_{t-k} and the correspondingly filtered output β_t at lags K = 0,1,2.... and given by

$$r_{\alpha\beta}(k) = \frac{c_{\alpha\beta}(k)}{s_\alpha s_\beta}$$

where

$$c_{\alpha\beta}(k) = \frac{1}{N} \sum_{j=1}^{N-k} (\alpha_j - \bar{\alpha})(\beta_{j+k} - \bar{\beta}) \qquad k = 0,1$$

$$s_\alpha = \sqrt{c_{\alpha\alpha}(0)}$$

$$s_\beta = \sqrt{c_{\beta\beta}(0)}$$

Impulse response weights which are equivalent to the cross correlation function are given by:

$$\hat{V}_k = \frac{s_\beta}{s_\alpha} r_{\alpha\beta}(k)$$

The cross correlation and impulse response are displayed. Knowing the V_k, values of the degrees r and s for the autoregressive and moving average operators and of the delay parameter b, may be estimated using the following rules [6]

Recalling the transfer function model (2.2.15)

$$Y_t = \delta^{-1}(B)\,\omega(B)\,X_{t-b} + N_t$$

The impulse response weights V_j consists of

(i) b zero values $V_o, V_1, \ldots V_{b-1}$

(ii) a further s-r+1 values $V_b, V_{b+1}, \ldots V_{b+s+r}$ following no fixed pattern.

(iii) values V_j with $j > b+s+r+1$ which follow the pattern dictated by an r-th order difference equation which has r starting values $V_{b+s}, \ldots V_{b+s+r+1}$. Starting values V_k for $j < b$ are clearly zero.

In addition the sample impulse response plots supplied by B-J experience and intution help to identify the vital parameters b, r and s .

A transfer function not only identifies the relationship between output y_t and input x_t, but also models the noise representing that part of the data which cannot be explained by the model, i.e. it represent the effect of all variables omitted from the model. Using the estimates V_x of the impulse response weights, the noise series n_t regenerated from

$$n_t = y_t - \hat{v}_o x_t - \hat{v}_1 x_{t-1} \cdots\cdots \hat{v}_g x_{t-g}$$

where g is the number of V_i weights for generating noise series n_t.

This is then treated as a univariate stochastic series and use the methods of ARIMA modeling for single series are employed to identify the noise process.

(ii) Estimation

After the order of transfer function has been determined the estimation procedure for model parameters follows closely that given for univariate ARIMA models. It begins with the preliminary estimates of the parameters. To estimate the autoregressive parts in the transfer function, for $r > 0$, the set of equations

$$A \hat{\delta}_o = h$$

are solved, where

$$A_{ij} = \begin{cases} \hat{v}_{b+s+i-j} & s+i \geq j \\ 0 & s+i < j \end{cases}$$

$$h_i = \hat{v}_{b+s+i}$$

$$i,j = 1,2,.....r$$

For the initial estimates of w_o of the moving average part of the transfer function

$$\hat{\omega}_{oo} = \hat{v}_b$$

If $r \geq s$,

$$\hat{\omega}_{jo} = \sum_{i=1}^{j} \hat{\delta}_i \hat{v}_{b+j-i} - \hat{v}_{b+j} \qquad (2.3.11)$$

If $r < s$,

$$\hat{\omega}_{jo} = \sum_{i=1}^{j} \hat{\delta}_i \hat{v}_{b+j-i} - \hat{v}_{b+j} \qquad j = 1,2,....r$$

$$\hat{\omega}_{jo} = \sum_{i=1}^{r} \hat{\delta}_i \hat{v}_{b+j-1} - \hat{v}_{b+j} \qquad j = r+1,....s$$

Maximum Likelihood estimates of preliminary estimates are obtained by minimizing the conditional sum of squares function

$$S_o(b, \delta, \omega, \phi, \theta) = \sum_{t=1}^{N} a_t^2 \ (\underline{b}, \delta, \omega, \phi, \theta \mid x_o, y_o, a_o)$$

(2.3.12)

The modified Gauss-Newton method of Appendix [A] can again be used for this non-linear estimation. It is clear that the number of parameters in a non-linear least squares routine increases considerably for transfer function analysis as compared to the univariate ARMA(n,m) case.

(iii) Diagnostic Checking:

The adequacy of a transfer function model is evaluated using two tests. (a) The first test is similar to the test performed for univariate time series. It involves examination of the autocorrelation function $\gamma_{\hat{a}\hat{a}}(k)$ of the residuals from the fitted model. The residual autocorrelations are obtained from,

$$r_{\hat{a}\hat{a}}(k) = c_{\hat{a}\hat{a}}(k) \,/\, c_{\hat{a}\hat{a}}(o) \qquad k = 0,1,.....k$$

where

$$(N - s - b - p - 1)\ c_{\hat{a}\hat{a}}(k) = \sum_{t-s+b+p+1}^{N-k} (\hat{a}_t - \hat{a})(\hat{a}_{t+k} - \hat{a})$$

(2.3.13)

The chi-square statistic is calculated from

$$Q = (n - s - b - p) \sum_{k=1}^{K} \gamma^2_{\hat{a}\hat{a}}(k)$$

and is compared with a chi-square distribution with K-p-q degrees of freedom, where p and q are the autoregressive and moving average order of the noise model.

(b) The second test involves checking the cross correlation function $\gamma_{\alpha\hat{a}}(k)$ between prewhitened input x_t and the residuals $\hat{a}_t$. The prewhitened input α_t is generated from (2.3.7) while the residuals are calculated from (2.2.15). Next cross correlation are given by(2.2.17):

The chi-square statistic is calculated from

$$Q = (n - v) \sum_{k=0}^{K} r^2_{\alpha\hat{a}}(k)$$

where

$$v = max\ (s + b + p, n)$$

and is compared with a chi-square distribution with K-r-s degrees of freedom.

2.4 Forecasting

One of the principal aims of modeling is prediction or forecasting. The manipulation or control of the behaviour is also based on prediction, so prediction may be viewed as the basis of control and regulation[30].

In many systems accurate forecasts can be obtained by using information derived from some related series. Such a related series is then called a "leading indicator"[6]. Treating the leading indicator series as X_t and the desired output series as Y_t these are related by the transfer function model obtained by the procedure outlined in the last section. The procedure in this section can then be used to accurately forecast Y_t by the leading indicator x_t.

It is assumed that an adequate transfer function model between series X_t and Y_t is

$$\phi(B)\,\delta(B)\,Y_t = \phi(B)\,\omega(B)\,X_{t-b} + \delta(B)\,\theta(B)\,a_t \qquad (2.4.1)$$

It is further assumed that an adequate stochastic model for the leading series is

$$X_t = \phi_x^{-1}(B)\,\theta_x(B)\,\alpha_t \qquad (2.4.2)$$

Equation (2.4.1) is written in the form

$$\delta^*(B)\,Y_t = \omega^*(B)\,X_{t-b} + \theta^*(B)\,a_t \qquad (2.4.3)$$

Now it is required to forecast a value X_{t+l} or Y_{t+l}, ℓ step ahead of the current time t. This forecast is said to be made at origin t for lead time ℓ. Using square brackets to denote conditional expectations at time t, and writing p* =p + d, we have for the lead-ℓ forecast,

$$\hat{Y}_t(\ell) = [Y_{t+\ell}] = \delta_1^* [Y_{t+\ell-1}] + \ldots . \delta_{p^*+r}^* [Y_{t+\ell-p^*-r}]$$

$$+ \omega_o^* [X_{t+\ell-b}] \ldots . - \omega_{p^*+s}^* [X_{t+\ell-b-p^*-s}] + [a_{t+\ell}]$$

$$- \theta_1^* [a_{t+\ell-1}] - \ldots \ldots \ldots \theta_{q+r}^* [a_{t+\ell-q-r}] \quad (2.4.4)$$

where

$$[Y_{t+j}] = \begin{cases} Y_{t+j} & j \leq 0 \\ \hat{Y}_{t(j)} & j > 0 \end{cases} \quad (2.4.5)$$

i.e. if j is less than or equal to zero the true value of series Y_t is used. But when j becomes greater than zero, the estimated or forecasted value Y_t (j) is obtained from equation (2.4.1), Also

$$[X_{t+j}] = \begin{cases} X_{t+j} & j \leq 0 \\ \hat{X}_{t(j)} & j > 0 \end{cases} \quad (2.4.6)$$

This is similar to the above except that the forecast or estimated value of $X_t(j)$ is calculated using the ψ_x weights, where

$$\psi_x(B) = \frac{\theta_x(B)}{\Phi_x(B)} \quad (2.4.7)$$

The updating of forecasts is achieved by

$$\hat{x}_{t+1}(\ell) = \hat{x}_t(\ell+1) + \psi_{x\ell} a_{x(t+1)} \quad (2.4.8)$$

Supposing that forecasts at origin t for lead time 1,2,.....L are available then as soon as x_{t+1} becomes available, a_t is updated by

$$a_{x(t+1)} = x_{t+1} - \hat{x}_t(1) \quad (2.4.9)$$

and is used to update forecasts from equation (2.4.8) at origin t + 1, for lead times 1,2,.....L-1, where 'L' is the farthest time ahead required for the forecast. The new forecast $x_{t+1}(1)$, for lead time L cannot be calculated by this means, but is easily obtained from forecasts at shorter lead times, using the difference equation.

Returning to the residuals from the transfer function

$$[a_{t+j}] = \begin{cases} a_{t+j} & j \leq 0 \\ 0 & j > 0 \end{cases} \quad (2.4.10)$$

a_t recursively calculated from equation (2.4.1), or if b $\geq$ 1, from

$$a_t = Y_t - \hat{Y}_{t-1}(1) \quad (2.4.11)$$

When subscript j is greater than 0 value of a_t is replaced with its expected value, i.e zero mean.

Thus by appropriate substitutions the forecasts for series Y_t are obtained from equation (2.4.4).

Forecasts are of little value unless they are accompanied by the respective probability limits. As a first step in calculating probability limits the ψ weights and the ν weights by

$$\delta(B)\ \phi_x(B)\ \nu(B) = \omega(B)\ \theta_x(B)\ B^b \qquad (2.4.12)$$

and

$$\phi(B)\ \psi(B) = \theta(B)$$

Then the variance of the lead-l forecast is then given by

$$V(\ell) = E\left\{Y_{t+\ell} - \hat{Y}_t(\ell)\right\}^2 = \sigma_\alpha^2 \sum_{j=b}^{\ell-1} \nu_j^2 + \sigma_a^2 \sum_{j=0}^{\ell-1} \psi_j \qquad (2.4.14)$$

Assuming that the a's are normal, it follows that, given information upto time t, the conditional probability distribution $P(Y_{t+1}|Y_{t-1}\ Y_{t-2}\cdots)$ of a future value Y_{t+1} of the process will be normal with mean $Y_t(\ell)$ and standard deviation equal to $V(\ell)$.

Therefore, approximate $100(\varepsilon)$ percent probability limits on the forecast for period t+1 may be computed from

$$\hat{y}_{t+\ell}(\pm) = \hat{y}_t(\ell) \pm u_{\varepsilon/2} \left\{V(\ell)\right\}^{\frac{1}{2}} \qquad (2.4.15)$$

where $u_{\varepsilon/2}$ is a $100(\varepsilon)$ percentile of the standard normal distribution. It is customary to choose value of ε equal to .95 or .50 giving 95% or 50% probability limits.

Chapter 3

PANDIT & WU TECHNIQUE

3.1. Justification for it's Use

As outlined in the last chapter Box and Jenkins provided an interactive, practical modeling technique which gives adequate models for forecasting and control. However despite its popularity, the Box and Jenkins method is not without problems and shortcomings.

One of the major problems arises in estimating the model orders n and m for a mixed ARMA (n,m) process. Although they have provided empirical guidelines for the determination of n and m when one of them is zero, in the mixed case their instructions for comparing graphs and charts often lead to inadequate models being fitted. This problem is more significant for higher order models.

The Box-Jenkins techniques are suitable for fitting ARMA models for both univariate time series and transfer function analysis. Their technique for the single-input single-output case is sound, but it is difficult to generalize it for multiple-input systems [13]. Taking the case of a two-input one-output system of the form

$$y(k) = \frac{\omega_1 (B)}{\delta_1 (B)} x_1 + \frac{\omega_2 (B)}{\delta_2 (B)} x_2 + \frac{\theta (B)}{\phi (B)} a_k \qquad (3.1.1)$$

Expanding it gives

$$y(k) = \omega_1(B)\ \delta_2(B)\ \phi(B)\ x_1(k) + \omega_2(B)\delta_1(B)\ \phi(B)\ x_2(k)$$

$$-\delta_1(B)\ \delta_2(B)\ \phi(B)\ y(k-1) + \delta_1(B)\ \delta_2(B)\ \theta(B)\ a(k)$$

This shows how many polynomial multiplications and divisions make the multi-input single-output problem complicated.

The fact that the method is a fully interactive technique gives the user the advantage of understanding the problem and diagnosing it easily. However this step-by-step process involving human interaction in every phase sometimes makes it cumbersome for the user, and the task becomes monumental when the user has many time series to analyze in a short time. So it was thought essential to find an alternative time series analysis technique which is automatic and requires minimum user interaction. Furthermore, programming an automatic technique will provide the opportunity to compare interactive and batch techniques. An important aspect to be studied in choosing the alternative technique for transfer function analysis of ARMA processes is that it should overcome the deficiencies of the Box & Jenkins method in finding n and m and it should be easily extendable to multi-input case.

To make a decision as to which technique to choose to supplement the Box and Jenkins method, several different techniques outlined in the literature survey were evaluated. The more recent technique of Pandit and Wu [31] was chosen to be the second time series analysis technique to be implemented on the microcomputer for this research.

According to Pandit and Wu [4] a mathematical model is at best the simplest representation of the behaviour of a given phenomenon, such that it cannot be significantly improved by more complicated ones, as judged by the observed data. Pandit and Wu's modeling strategy explicitly formulates modeling as a successive improvement problem. Unlike the existing approaches, it does not assume that there is some fixed model generating the data and does not formulate the modeling problem as that of identification, of this model by a more or less game theoretic approach.

Pandit and Wu's modeling strategy is to approximate the dependence in the data more and more closely by an increasing sequence of ARMA(n,n-1) models. The stopping criterion is the value of n beyond which no significant improvement in the approximation, as judged by the reduction in the residual sum of squares, is observed. This basic idea results in a simple modeling procedure that can be completely executed automatically on a digital computer. In other words, once the data is supplied to the modeling routine, it can output all

of the adequate models pointing out the most adequate one; no empirical examination of plots of autocorrelation, spectra, residuals, etc. is required nor is any preprocessing such as differencing, filtering, etc. required in dealing with non-stationary data.

The data that are obtained from the system being identified by the Pandit and Wu technique may be univariate or multivariate. The fact that the same modeling strategy is applicable to both the univariate and the multivariate data is also of great significance in the development of multivariate time series analysis for transfer function identification. Other methods of transfer function are generally restricted to the bivariate case only. Often, identification of more than two series of data [4] is both time consuming and laborious, but with the present method, when employed in conjunction with the checking criteria, a large number of input-output data can be handled with relative ease. Moreover little difficulty is encountered in fitting higher-order models by the Pandit and Wu technique. But due to the mathematical intensiveness of the technique limitations are imposed by computer capability and the efficiency of the non-linear routine.

The application of the Pandit and Wu technique in practice has been encouraging. Its methodology is applicable to diverse problems. While there has been a concentration of

these applications in areas pertaining to production engineering, partly because the technique evolved from the Mechanical Engineering Department at the University of Wisconsin, its application has also been extended to systems in different fields such as the blast furnace, large scale systems and on-line safety control[32]. A complete up-to-date list of the applications of the technique can be found in [4].

3.2 General Model

The autoregressive moving average model for a stationary zero-mean univariate time series is of the form

$$X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} - \ldots\ldots - \phi_n X_{t-n} \qquad (3.2.1)$$

$$= a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \ldots\ldots - \theta_{n-1} a_{t-n+1}$$

$$E[a_t] = 0 \; ; \quad E[a_t \, a_{t-k}] = \delta k \, \sigma_a^2$$

where δ_k is the Kronecker delta function, ϕ_1, ϕ_2,.. ϕ_n are autoregressive parameters, θ_1 θ_2,... θ_{n-1} are moving average parameters; and the a_t's are normally distributed with zero mean and σ_{a^2} variance.

The ARMAV or vector form for a multivariate time series

$$X_t = (X_{1t}, X_{2t}, \ldots\ldots\ldots X_{pt})$$

is

$$X_t = \Phi_1 X_{t-1} + \Phi_2 X_{t-2} + \ldots\ldots + \Phi_n X_{t-n} + \alpha_t \qquad (3.2.2.)$$

$$- \Theta_1 \alpha_{t-1} - \Theta_2 \alpha_{t-2} - \ldots\ldots\ldots - \Theta_m \alpha_{t-m}$$

$$E[\alpha_t, \alpha_{t-k}] = \delta k \sigma_\alpha^2 \qquad E[\alpha_t] = 0$$

where X_t and α_t are px1 column vectors and

$$\Phi_k = \{\phi_{kij}\} \ , \ k = 1,2,\ldots\ldots\ldots n$$

$$\Theta_k = \{\theta_{kij}\} \ , \ k = 1,2,\ldots\ldots\ldots n-1$$

are pxp matrix autoregressive and moving average parameters. The pxp matrix $\sigma_{\alpha^2} = (\sigma_{a_{ij}})$ is the covariance matrix of α_t.

As an example, for a bivariate series with p = 2, the ARMAV(2,1) model is

$$\begin{pmatrix} X_{it} \\ X_{2t} \end{pmatrix} = \begin{pmatrix} \phi_{111} & \phi_{112} \\ \phi_{121} & \phi_{122} \end{pmatrix} \begin{pmatrix} X_{it-1} \\ X_{2t-1} \end{pmatrix} + \begin{pmatrix} \phi_{211} & \phi_{212} \\ \phi_{221} & \phi_{222} \end{pmatrix} \begin{pmatrix} X_{it-2} \\ X_{2t-2} \end{pmatrix}$$

$$+ \begin{pmatrix} a_{1t} \\ a_{2t} \end{pmatrix} - \begin{pmatrix} \theta_{111} & \theta_{112} \\ \theta_{121} & \theta_{122} \end{pmatrix} \begin{pmatrix} a_{it-1} \\ a_{2t-1} \end{pmatrix} \qquad (3.2.3)$$

The conditional sum of squares and products (CSP) matrix is

given by

$$A = \begin{bmatrix} \Sigma a_{1t}^2 & \Sigma a_{1t}a_{2t} \\ \Sigma a_{1t}a_{2t} & \Sigma a_{2t}^2 \end{bmatrix}$$

The covariance σ_{a^2} is estimated by (1/N).A where N is the number of vector observations.

3.3 Data Characteristics

3.3.1 Green's Function

Traditionally, the autocovariance function has played a major role in the development of the ARMA model as undertaken by statisticians. However, according to Pandit and Wu, for physical systems, more information about the system dynamics is obtained from the Green's function (or impulse response).

Green's Function of the ARMA(2,1) System

Consider an ARMA(2,1) model for equation (3.2.1)

$$X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} = a_t - \theta_1 a_{t-1}$$

or

$$(1 - \phi_1 B - \phi_2 B^2)\, X_t = (1 - \theta_1 B)\, a_t$$

Let the autoregressive part of the ARMA(2,1) model be factorized as

$$(1 - \phi_1 B - \phi_2 B^2) = (1 - \bar{\lambda}_1 B)(1 - \lambda_2 B)$$

or

$$\lambda_1 + \lambda_2 = \phi_1$$

$$\lambda_1\lambda_2 = -\phi_2$$

where λ_1 and λ_2 are the characteristic roots of the second linear difference equation given by

$$\lambda^2 - \phi_1\lambda - \phi_2 = 0 \qquad (3.3.1)$$

i.e.

$$\lambda_1, \lambda_2 = \frac{1}{2}\left(\phi_1 \pm \sqrt{\phi_1^2 + 4\phi_2}\right)$$

Wold's decomposition of the ARMA(2,1) process X_t is given by

$$X_t = \frac{(1 - \theta_1 B)a_t}{(1 - \phi_1 B - \phi_2 B^2} = \frac{(1 - \theta_1 B)a_t}{(1 - \lambda_1 B)(1 - \lambda_2 B)}$$

Assuming the two roots are distinct, this expansion can be obtained using partial fractions

$$X_t = \sum_{j=0}^{K} \left[\left(\frac{\lambda_1 - \theta_1}{\lambda_1 - \lambda_2}\right)\lambda_1^j + \left(\frac{\lambda_2 - \theta_1}{\lambda_2 - \lambda_1}\right)\lambda_2^j\right] a_{t-j}$$

Definition: Green's function describes the memory or dynamics of a system in terms of past a_t's and shows how they affect the response X_t. It is obtained by expressing X_t as a linear combination of the a_t's. The boundedness of the Green's function imposes stability conditions on the autoregressive

parameters of the ARMA models.

Based on this definition Green's function for an ARMA(2,1) model is

$$G_j = \left(\frac{\lambda_1 - \theta_1}{\lambda_1 - \lambda_2}\right)\lambda_1^j + \left(\frac{\lambda_2 - \theta_1}{\lambda_2 - \lambda_1}\right)\lambda_2^j \qquad (3.3.2)$$

Physical Interpretation

It is clear from the above expansion that G_j is the weight, given in the present response, to the shock or disturbance a_t, which entered the system j time units earlier. The quantity G_j indicates how well the system remember the shocks a_{t-j}.

Moreover Green's function G_j characterizes how slowly or rapidly dynamic response of the system to any particular a_t decays. In other words, if a single a_t is injected into the system, Green's function determines how quickly the system will return to its equilibrium or mean position, which is always taken to be zero.

Green's Function of the ARMA (n,n-1) System

Repeating the partial fraction expansion for distinct roots for the ARMA(n,n-1) model, it has been shown [4] that Green's function has the form:

$$G_j = g_1 \lambda_1^j + g_2 \lambda_2^j + \ldots\ldots\ldots g_n \lambda_n^j \qquad (3.3.3)$$

$$X_t = \sum_{j=0}^{\infty} G_j a_{t-j} \qquad 80.$$

where $\lambda_1, \lambda_2 \ldots\ldots \lambda_n$ are the roots of

$$\lambda^n - \phi_1 \lambda^{n-1} - \phi_2 \lambda^{n-2} \ldots\ldots - \phi_n = 0 \qquad (3.3.4)$$

and

$$g_i = \frac{(\lambda_i^{n-1} - \theta_1 \lambda_1^{n-2} - \ldots\ldots \theta_{n-1})}{(\lambda_i - \lambda_i)(\lambda_i - \lambda_2)\ldots(\lambda_i - \lambda_{i-1})(\lambda_i - \lambda_{i+1})\ldots(\lambda_i - \lambda_n)}$$

$$i = 1,2\ldots n \qquad (3.3.5)$$

where the denominator is the product of all terms $(\lambda_i - \lambda_j)$ for j=1,2,....n excluding the zero term $(\lambda_i - \lambda_i)$.

Stability Conditions

The system is asymptotically stable if for values of λ_j, given sufficient time, the system asymptotically returns to its equilibrium position if only one a_t is injected. The conditions for an ARMA(n,m) system are

$$|\lambda_k| < 1 \quad , \quad k = 1,2,\ldots\ldots n \qquad (3.3.6)$$

If $|\lambda_k| = 1$, $G_j = 1$ and a single a_t is injected, the system stays at position a_t indefinitely, then it is not asymptotically stable. But since the response at subsequent time will not exceed a_t (i.e. it will remain bounded), the system will be stable. So the stability conditions for an

ARMA(n,m) system are

$$|\lambda_k| \leq 1 \quad , \quad k = 1,2,.....n$$

and if

$$\lambda_i = \lambda_j \ , \ i \neq j \ \text{ then } \ |\lambda_i| = |\lambda_j| < 1 \qquad i,j = 1,.....n$$

Green's Function was used by Miller in 1968 [4]. Weiner used it much before (1949) and called it a weighting function. Box & Jenkins called the elements "psi weights".

3.3.2 Inverse Function

This function describes the dynamics or memory in terms of the influence of past X_t's on the present X_t by decomposing X_t as a linear combination of past X_t's. It is known as the inverse function since it is obtained by inverting the operator that yields Green's function. Denoted by I_j, the boundedness of the Inverse function imposes "invertibility" restrictions on the moving average parameters of the ARMA model; which are similar to the stability conditions discussed in the last section. Define

$$X_t = \sum_{i=1}^{\infty} I_i \, X_{t-j} + a_t \qquad (3.3.7)$$

or

$$a_t = (1 - I_1 B - I_2 B^2 .....) X_t$$

Inverse Function of the ARMA(1,2) System

Consider an ARMA(1,2) model for equation (3.2.1)

$$X_t - \phi_1 X_{t-1} = a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2}$$

or

$$(1 - \phi_1 B) X_t = (1 - \theta_1 B - \theta_2 B^2) a_t$$

Factoring the moving average part of the ARMA(1,2) model as

$$(1 - \theta_1 B - \theta_2 B^2) = (1 - \nu_1 B)(1 - \nu_2 B)$$

that is

$$\nu_1 + \nu_2 = \theta_1$$

$$\nu_1 \nu_2 = -\theta_2$$

where ν_1 and ν_2 are the characteristic roots of the moving average operator

$$\nu^2 - \theta_1 \nu - \theta_2 = 0$$

so that

$$\nu_1, \nu_2 = \frac{1}{2} (\theta_1 \pm \sqrt{\theta_1^2 + 4\theta_2}$$

The equations (3.3.8) obtained so far are similar to

equations (3.3.1) given in the last section, the only difference being that the role of autoregressive and moving average parameters has been interchanged. Using the same set of procedures used for finding G_j, interchange ϕ and θ, λ and ν and by change of sign when roots are distinct,

$$I_j = -\left(\frac{\nu_1 - \phi_1}{\nu_1 - \nu_2}\right)\nu_1^{\,i} - \left(\frac{\nu_2 - \phi_1}{\nu_2 - \nu_1}\right)\nu_2^{\,i}$$

This equation is same as equation (3.3.2) for G_j. This duality between the Green's function and inverse function is shown by the block diagram Fig 3.1.

Inverse Function of the ARMA(n,n-1) models

From the ARMA(1,2) model the inverse function for a general ARMA(n,n-1) model may be found by using equations (3.3.3) (3.3.4) and replacing,

$$G_j \rightarrow -I_j$$

$$\lambda_k \rightarrow \nu_k$$

$$g_i \rightarrow i_j$$

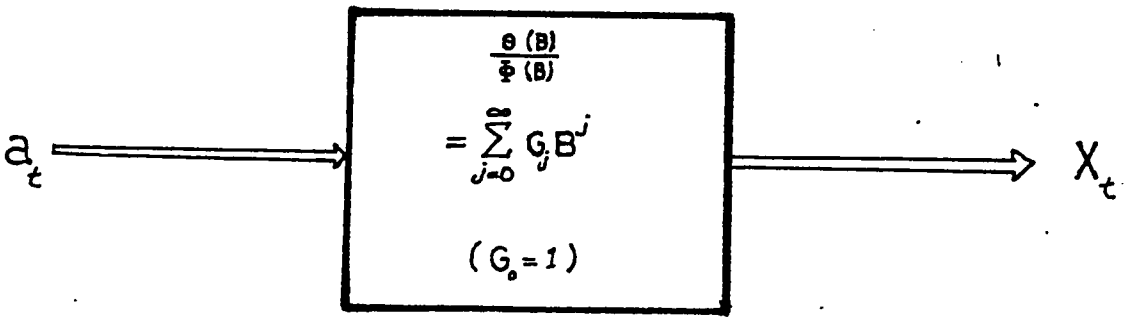
$$n \rightarrow m$$

$$\phi \rightarrow \theta$$

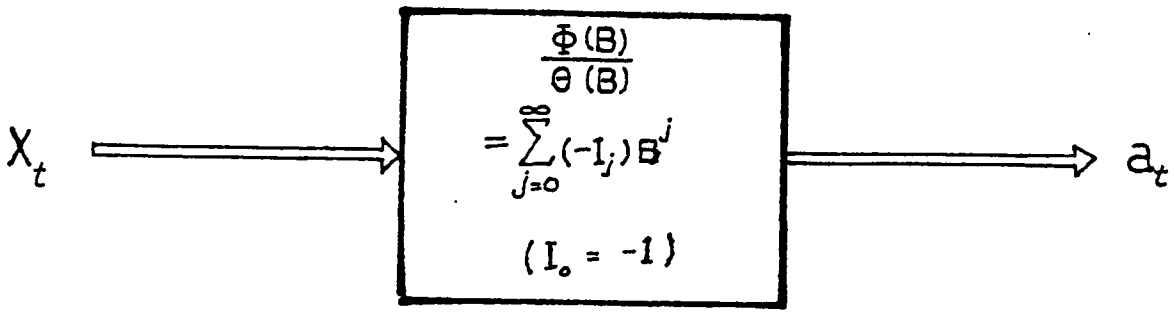
this gives

$$I_i = -(i_1\nu_1^{\,j} + i_2\nu_2^{\,j} + \ldots\ldots i_m\nu_m^{\,j}) \qquad (3.3.9)$$

where $\nu_1, \nu_2 \ldots \nu_m$ are the roots of



a) Green's Function



b) Inverse Function

Fig. 3.1 Relationship between Green's Function and Inverse Function

$$\nu_1^m - \theta_1 \nu_1^{m-1} - \theta_2 \nu^{m-2} \ldots\ldots - \nu_m = 0$$

and

$$i_k = \frac{(\nu_k^{m-1} - \phi_1 \nu_k^{m-2} \ldots\ldots - \phi_{m-1})}{(\nu_k - \nu_1)(\nu_k - \nu_2)\ldots\ldots(\nu_k - \nu_{k-1})(\nu_k - \nu_{k+1})\ldots(\nu_k - \nu_m)}$$

$$k = 1,2,\ldots\ldots m \qquad (3.3.10)$$

Invertibility and its Applications

If the roots ν_k are greater than one in absolute value, then the inverse function increases without bound. This implies that the more distant in the past, the greater the influence of the past X_t's on the present one. Such a situation is physically meaningless and therefore an 'invertibility' condition is usually imposed on the ARMA models. The invertibility conditions for a general ARMA (n,m) model are

$$|\nu| < 1 , \quad k = 1,2,\ldots\ldots m \qquad (3.3.11)$$

The inverse function is used to estimate the initial values of the parameters needed to start the non-linear least squares routine. Here it is necessary to recursively compute the a_t's at each iteration for computing the sum of squares

of the a_t's. Using eq.(3.3.7)

$$a_t = X_t - \sum_{j=1}^{\infty} I_j \; X_{t-j}$$

If the moving average parameters θ_i's do not satisfy invertibility conditions then I_j and a_t respectively tend to diverge. Choosing the initial values of θ_i within the invertibility limits solves this problem.

3.4 ARMA (n,n-1) Models

Data collected for modeling a system can be represented (or approximated) by a linear differential equation. The question then is: What order of approximation will be adequate? Pandit and Wu have tacked this problem by fitting higher order (n,n-1) models and applying checks until an adequate model is obtained. Pandit [26] states a fundamental theorem:

> "Every stationary process can be approximated by some ARMAV process and for purpose of approximation, processes with a moving average order less than the autoregressive order are enough".

ARMA(n,n-1) modeling is convenient when higher order

models are desired. Sometimes in ARMA(n,m) modeling it may not be possible to guess the orders m and n, or the guessed order may turn out to be wrong after fitting. This leads to trial and error. A rational way in which to carry out trial and error is to successively fit all models of orders less than one, two and so on.

Supposing that an ARMA(5,3) model is required for the data, then as many as 24 models could be fitted before reaching the adequate model, whereas fitting by ARMA(n,n-1) models would require only five iterations before the adequacy is obtained [4]. Also the five models in the (n,n-1) approach would automatically, be checked by the algorithm in section whereas 24 models in the above ARMA(n,m) models would each have to be individually checked.

The rationale of the ARMA(n,n-1) modeling strategy is most easily seen by considering G_j or γ_k given by (3.3.3)[18]

$$G_j = g_1 \lambda_1^j + g_2 \lambda_2^j + \ldots . g_n \lambda_n^j \qquad (3.4.1)$$

$$\gamma_k = d_1 \lambda_1^k + d_2 \lambda_2^k + \ldots \ldots d_n \lambda_n^k$$

where $\lambda_1, \lambda_2 \ldots \lambda_n$ are the characteristic roots of the

autoregressive part. The problem of finding a model for which the above equations (3.4.1) are the Green's function and the autocovariance function respectively is the same as finding a homogeneous difference equation with suitable initial conditions for which they are the solutions. The n linearly independent roots $\lambda_1, \lambda_2 \ldots \lambda_n$ determine the autoregressive order n, and autoregressive part given by

$$(1 - \phi_1 B - \phi_2 B^2 - \ldots\ldots \phi_n B^n)\ G_j = 0 \qquad j \geq 0$$

$$(1 - \phi_1 B - \phi_2 B^2 - \ldots\ldots \phi_n B^k)\ \gamma_k = 0 \qquad k \geq n$$

The relation between λ_i's and ϕ_i's is given by

$$(1 - \lambda_1 B)(1 - \lambda_2 B)\ldots\ldots(1 - \lambda_n B)$$

$$= (1 - \phi_1 B - \phi_2 B^2 \ldots\ldots \phi_n B^n)$$

The n constants $g_1, g_2, \ldots\ldots g_n$ or $d_1, d_2, \ldots\ldots d_n$ have to be specified by n initial conditions, but one restriction must always hold

$$1 = G_o = g_1 + g_2 + \ldots\ldots g_n$$

or

$$\gamma_o = d_1 + d_2 \ldots\ldots\ldots\ldots d_n$$

Hence, only n-1 conditions remain to be specified, which

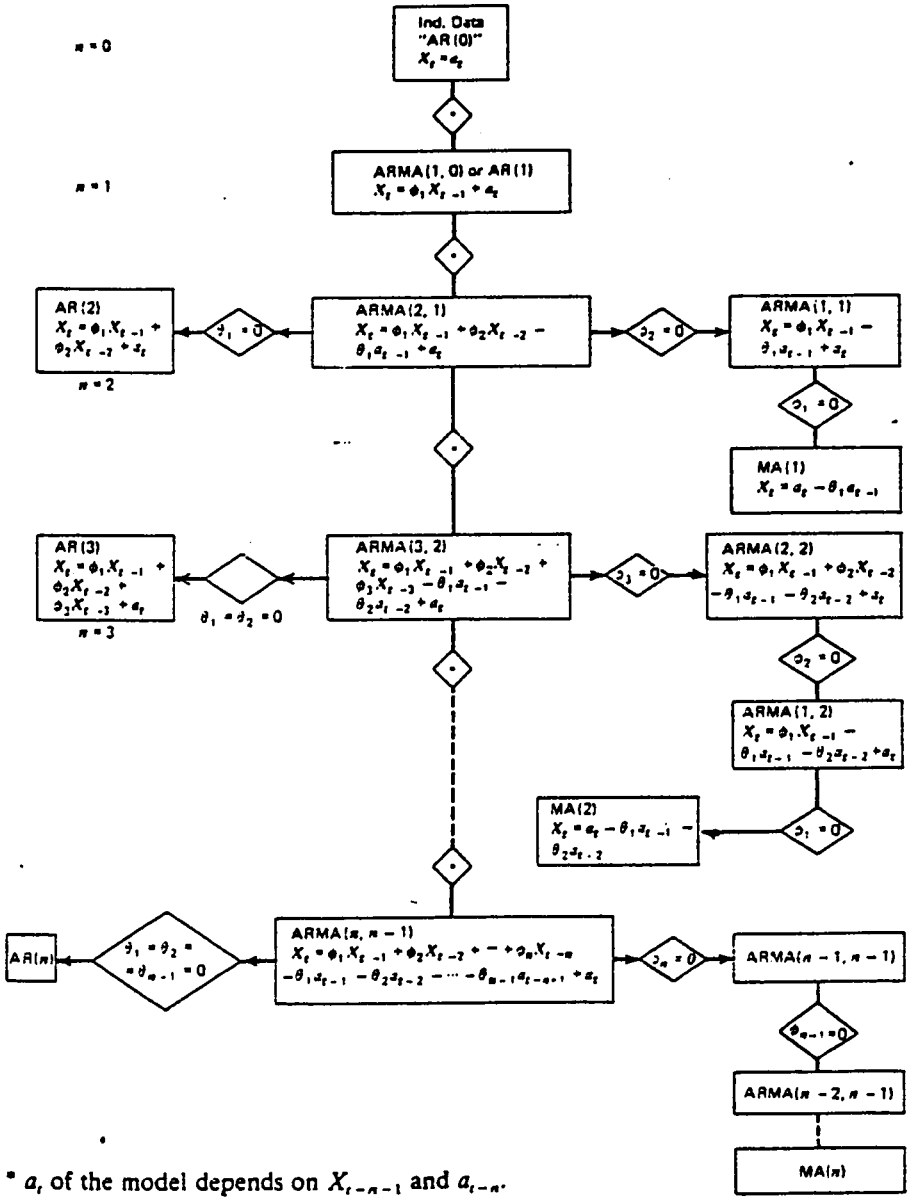
determine the n-1 moving average parameters $\theta_1, \theta_2, \ldots \theta_{n-1}$ and this gives the ARMA(n,n-1) model.

At a first glance it appears that the class of ARMA(n,n-1) models is too narrow and does not include such well known models as AR(2),MA(1) etc. This is not correct. The AR(2) model is the same as the ARMA(2,1)with $\theta_1 = 0$.

The ARMA(n,n-1) model obtained by the above strategy suffices for most of the practical applications such as forecasting, control, stability analysis etc. However, if one desires to further reduce the number of parameters and to see whether some of the special cases of the ARMAV(n,n-1) models may also be adequate, useful clues are provided by the confidence interval of the parameters. The evolution of ARMA(n,n-1) models starting from the independence assumption, or AR(0) model, is schematically presented in Fig. 3.2. It shows how ARMA(n,n-1) models of increasing order arise. All special cases including the pure autoregressive and pure moving average models are shown.

3.4.1 ARMA (2n,2n-1) Model

Although the above procedure is feasible in theory and is known to work well for small data sets (number of observation around 200) and small model order (around n



* a_t of the model depends on X_{t-n-1} and a_{t-n}.

Fig. 3.2 Genesis of ARMA Models

= 4), increasing n by one is not economical in practice. Empirical and practical experiences have indicated that it is more feasible to increase in steps of two and fit ARMA(2n,2n-1) models for n = 1,2,3..... Pandit and Wu [4] have advocated two reasons for employing this approach.

Firstly, it has been observed that when modeling data sets from specific experiments the degrees of freedom play an important role. Taking an example from vibration systems, it was noted that for an experiment with one degree of freedom an ARMA(2,1) model was found to be adequate and for two degrees of freedom an ARMA(4,3) model was satisfactory. Thus increasing the degree of freedom by one amounts to advancing the autoregressive order by two. Hence an ARMA(2n,2n-1) model would fit more appropriately in these cases.

The second reason is related to the configuration of the characteristic roots λ_i, which may be real or complex. Complex roots always occur in conjugate pairs, so for an ARMA(2,1) model a complex root implies that its conjugate must be the second root. This applies for all ARMA models, and it may be concluded also that if the autoregressive order is odd, one of the roots must be real.

Given an ARMA(2,1) model and assuming that ARMA(4,3) with two conjugate pairs is the adequate model, then increasing the model order by one to ARMA(3,2) forces the third root to be real and hence it gives a poorer approximation. This problem can be avoided by increasing the model order by two to ARMA(4,3) from ARMA(2,1). For the case where ARMA(3,2) is the correct model then this would be indicated in the confidence intervals of the ARMA(4,3) model.

Thus when higher-order models are required, increasing the model order by two is more economical as only half the number of models are fitted. Moreover an added advantage of utilizing the ARMA(2n,2n-1) sequence is that it starts with the important ARMA(2,1) model, which is applicable to a large number of practical systems,so that it may often provide an adequate model at once.

3.5 Transfer Function Modeling Strategy

The theory of Pandit and Wu starts from a single series of data, considered as a realization of a stationary stochastic system and which can always be represented by a model of the ARMA form. The ARMA models can be extended to vector systems, which are analogous

to transfer function models, to show that two or more sets of data, treated as the realization of a vector stationary system, can be similarly represented by vector or ARMAV models. This work is concerned with multiple time series for application of transfer function analysis in forecasting and control, and hence the model is developed for the autoregressive moving average vector. For sake of completeness the modeling procedure is considered firstly for a single time series and is then extended to the vector case.

3.5.1 ARMA MODEL

The static and dynamic characteristics developed in the preceding section are employed here to evolve a simple and rational procedure for modeling ARMA systems from time series data.

i) Estimation

Since the modeling procedure consists of approximating the data successively by ARMA(n,n-1) models, increasing in steps of two, there is no necessity to select the model order. Starting from n=1,2,.... for ARMA(2n,2n-1), the first step required is the estimation of 2n ϕ_i and 2n-1 θ_i parameters. When a moving average parameter is present, the unconditional regression is non-linear and therefore the non-linear

least square method be used. The initial values of the parameters needed to start the nonlinear search routine can be obtained from the inverse function of the ARMA(n,m) model.

The ARMA(n,m) model(3.2.1) can be written as,

$$(1 - \phi_1 B - \phi_2 B^2 \ldots\ldots \phi_n B^n) \; X_t \qquad (3.5.1)$$

$$= (1 - \theta_1 B^1 - \theta_2 B^2 \ldots\ldots \theta_m B^m) \; a_t$$

Recalling equation (3.3.7)

$$a_t = (1 - I_1 B - I_2 B^2 \ldots\ldots) \; X_t \qquad (3.5.2)$$

Substituting for a_t from Eq.(3.5.2) in Eq.(3.5.1), gives the operator identity

$$(1 - \phi_1 B - \phi_2 B^2 - \ldots\ldots \phi_n B^n)$$

$$\equiv (1 - \theta_1 B - \theta_2 B^2 \ldots\ldots \theta_m B^m)(1 - I_1 B - I_2 B^2)$$

Equating the coefficients of equal powers of B, gives the expression

$$\phi_j = \theta_j - \theta_1 I_{j-1} - \theta_2 I_{j-2} \cdot\cdot\cdot\cdot\cdot \theta_{j-1} I_1 + I_j$$

for all j with the assumption that $\theta_j = 0$ for $j > m$ and $\phi_j = 0$ for $j > n$ for ARMA(n,m) models. In particular for

$j > \max(n,m)$

$$(1 - \theta_1 B - \theta_2 B^2 - \theta_3 B^3 \ldots\ldots - \theta_m B^m)\ I_j = 0$$

Good estimates of the I_j's can be obtained by truncating the infinite series in Eq.(3.3.7) as

$$X_t = I_1 X_{t-1} + I_2 X_{t-2} + \ldots\ldots + I_p X_{t-p} + a_t \qquad (3.5.3)$$

for large enough p and then estimating the I_j's by linear least squares. Using these $p > (n+m)$ I_j's gives the initial values of ϕ_j and θ_i by linear least squares.

Now with n and m initial value estimates of ARMA(n,m) model available, parameters of the model may be estimated using any of the standard library programs for non-linear least squares.

The non-linear least square routine first computes the a_t's or the "residual errors" recursively by

$$a_t = X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} \ldots\ldots \phi_n X_{t-n}$$

$$+ \theta_1 a_{t-1} + \theta_2 a_{t-2} + \ldots\ldots + \theta_m a_{t-m}$$

The routine then monitors these values in the direction of the smaller sum of squares of a_t's. Once a point in the parameter space giving a smaller sum of squares is

reached, it starts a new iteration with this point as the initial values. Th iterations continue until some specified tolerances are reached, such as the relative reduction in the sum of squares, or the maximum change in the parameter value is below, say, 10^{-5}, or the number of iterations is greater than, say 25. After the minimum has been attained the approximate confidence intervals on the estimated parameters are obtained by linear least squares theory, using the local linear hypothesis.

ii) Test of Adequacy

The final step in the Pandit and Wu technique is to decide the value of n for which the ARMA(n,n-1) model can be considered to be adequate. To answer this question Pandit and Wu have evolved the argument that the independence of a_t's for ARMA(n,n-1), guarantees the adequacy of an ARMA(n,n-1) model. The check on the independence of the a_t's is simple and reliable. The procedure is to fit a higher order model, and if the higher order model gives a substantial reduction in the residual sum of squares, it automatically implies that the a_t's of the present model are not adequate. On the other hand, if the present model is adequate and a higher order model is obtained then the reduction int he sum of squares will be smaller and in some cases it

might even be slightly larger. If unnecessary additional terms are being injected in the model, the presence of zero in the confidence interval of those parameters will show this and a lower order fit may be attempted.

Checking for independence of the a_t's is usually formulated as the hypotheses

H_o : a_t's are independent

H_2 : a_t's are dependent

This test, which is also the test of ARMA(2n,2n-1) versus ARMA(2n+2,2n+1), is achieved using the Wilks-Lambda test shown in Rao [4].

Test Statistic:

$$F = \frac{A_1 - A_o}{s} \div \frac{A_o}{N-r} \qquad (3.5.4)$$

Distribution

$$F\ (s,\ N - r)$$

For the test of ARMA (2n,2n-1) versus ARMA(2n+2,2n+1) the above notations can be defined as

r : 4n + 4

S : 4

A_o : sum of squares of a_t's for the

ARMA(2n+2,2n+1)model

A_1 : sum of squares of a_t's for the ARMA(2n,2n-1) model

N : number of observations

If the value of the F-distribution at say 5% significance level exceeds the value obtained from the test statistic, then the decrease in residual sum of squares going from the ARMA(2n,2n-1) model to the ARMA(2n+2,2n+1) is significant, and there is strong evidence to reject the hypothesis H_o - i.e. the a_t's are independent for the present ARMA(2n,2n-1) model. If the test statistic is less than the value obtained from the F-distribution tables then hypothesis H_o is accepted and it is concluded that the model is adequate at that level of significance.

Once the critical decision is made to accept an ARMA(2n,2n-1) model, this is taken as a base point in a search involving possible reduction of autoregressive or moving average order. Clues in this search will be provided by the presence of zero in the confidence intervals of any of these parameters. Based on these clues new assumptions are made about the model order and tested using the above F-criterion. Any requirement for special models to be fitted, such as pure autoregressive or pure moving average, can be accomplished in this same

way.

Two other checks suggested by Pandit and Wu for checking the adequacy of model order have been discussed in Chapter 2. One of the checks to test the independence of residuals is the requirement that residual autocorrelations fall between standard error limits of $\pm 2/\sqrt{N}$. Analytically this behaviour can also be tested by Box-Pierce chi square statistic

$$Q = N \sum_{k=1}^{k} \hat{\rho}_k^2 \qquad (3.5.5)$$

which should be less than $\chi^2_{(K-n-m)}$ at the appropriate significance level.

iii) Univariate ARMA Algorithm

Step 1

Fit the ARMA(2n,2n-1) model, starting with n=1. For every increase of n by one, check the improvement in the residual sum of squares of a_t's by the F-criterion, Eq.(3.4.2). Confirm that the autocorrelations of a_t's are within the permissible band $\pm 2/\sqrt{N}$. Stop as soon as the F value from ARMA(2n,2n-1) to ARMA(2n+2,2n+1) is insignificant at a predetermined value such as 5% and choose the ARMA(2n,2n-1) model.

Step 2

Check the values of ϕ_{2n}, θ_{2n-1} to see if they are small compared to their largest absolute value one and if their confidence intervals include zero. If not, the model ARMA(2n,2n-1) is adequate.

Step 3

If ϕ_{2n} and θ_{2n-1} are small and their confidence intervals include zero, fit an ARMA(2n-1,2n-2) model and compare it with an ARMA(2n,2n-1) by the F-criterion. If the F value is not significant, then dropping the small MA parameters, fit an ARMA(2n-1,m) model with $m < 2n-2$ and check by the F-criterion until the adequate model with the smallest number of parameters is reached.

Step 4

If the F value is significant, drop the small moving average parameters and determine an ARMA(2n,m) model with $m < 2n-1$), as in Step 3.

Step 5

Special models such as pure AR(n) ot pure MA(m) etc. can be fitted by checking each increasing model with the F-criterion and stopping at an insignificant reduction in residual sum of squares.

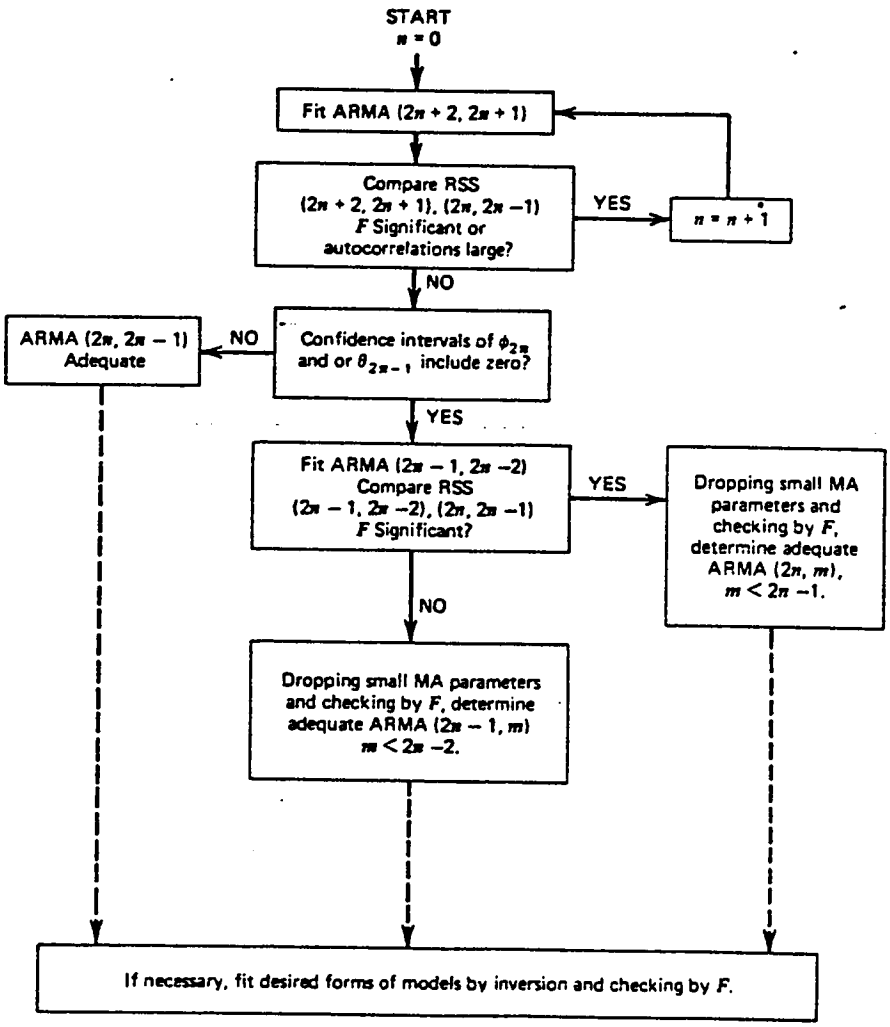


Fig. 3.3 ARMA(2n, 2n-1) Modeling Algorithm

The algorithm is shown in Fig 3.3.

3.5.2 Auto Regressive Moving Average Vector

Having described the modeling procedure for the ARMA(n,n-1) models, this is now extended to ARMAV(n,n-1) models. These models are helpful for developing strategies for forecasting and control. However before developing the vector models, it is appropriate to discuss the relative merits of two equivalent control system formulations: classical and modern control theory.

"Classical control theory" is based on the concepts of transfer functions and block diagrams , employing transform techniques such as Fourier, Laplace or Z-transforms. Classical control theory was specifically related to systems such as servomechanisms. The transfer function approach is intuitively appealing and easily understood by aid of block diagrams. Another advantage of the transfer function approach is that it is well suited for single-input single-output systems.

The form of the transfer function is generally known or estimated by physical considerations. The discrepancy between the output given by the transfer function alone and the actual output is then termed noise. In many cases, the physical source of noise can be clearly

identified as separate from the system represented by the transfer function. As transfer function is derived from physical reasoning and the noise source after the transfer from input to output is clearly identified. This is the approach basically adopted by Box and Jenkins. But transfer function analysis is cumbersome and complicated for multivariable systems.

In general, however, the noise disturbance as well as the transfer of input into output occurs or are distributed all over the system; hence transfer and noise models and their representations are mostly fictitious.

For this reason and also for other theoretical and computational advantages, "modern control theory" uses a state space or state variable approach in preference to the transfer function methodology. It places more emphasis on mathematical programming for computer calculation. The state space approach itself consists of treating all inputs and outputs (as well as their derivatives and differences if necessary) as the states of the system and forming a single vector with each of the states as its elements. The state variable may be observable (such as input, output, etc.) on unobservable (such as disturbances or derivative of input-ouput, etc.). This formulation is particularly

suitable for multivariable systems. Although these vector models lose their intuitive appeal, they are easy to treat, they are particularly simple for the purpose of modeling and are almost indispensable for systems with a large number of inputs and outputs.

The ARMA models for univariate time series discussed in earlier section are actually state variable models. For example, the discrete time autoregressive AR(1) model relates the observed states X_t and X_{t-1} with unobserved state a_t. Therefore, a natural extension of the ARMA models to control systems involving one or more inputs and one output is easily applicable. This model can always be converted into transfer function form. The ARMAV(n,n-1) forms were comprehensively studied by Pandit [33]. The method is based on successive orthogonalization. In the detailed theory of the approach [33], where cases of MIMO systems with large numbers of variables or large lags are discussed, together with singular multivariate autoregressive models proposed for this works form a special case of the generalized ARMAV models. This special case is termed an Extended Auto Regressive Moving Average Model (EARMA). Such models are adequate to deal with one output and one or two input systems.[4]

Recalling the ARMA(n,m) or EARMAV(n,m) model has the

form

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots\ldots \phi_n X_{t-n}$$

$$+ a_t - \theta_1 a_{t-1} \ldots\ldots \theta_m a_{t-m}$$

The variance-covariance matrix of a_t.

It should be noted that the basic assumption on a_t for the structure of the ARMAV models does not guarantee that a_{1t}, a_{2t} at simultaneous time t are independent or uncorrelated. Therefore, $E[a_{1t}, a_{2t}]$ may or may not be zero. For bivariate series, the variance-covariance matrix of a_t can be written as

$$\gamma_a = \begin{bmatrix} \gamma_{a11} & \gamma_{a12} \\ \gamma_{a21} & \gamma_{a22} \end{bmatrix}$$

where γ_{a11} and γ_{a22} denote the variance of a_{1t} and a_{2t}, and γ_{a12} and γ_{a21} represent their covariance.

In the case of EARMA models, certain broad assumptions are made. The usual assumption in control theory that the input noise (or the input itself) is independent of the output noise, implies that modeling and control can be accomplished by considering the two models separately, rather than in vector form. An EARMA(2,1) model for single input single output system can be written as

$$X_{1t} = \phi_{111} X_{1t-1} + \phi_{121} X_{1t-1} + a_1 t - \theta_{111} a_{1t-1}$$

$$X_{2t} = \phi_{211} X_{2t-1} + \phi_{221} X_{2t-1} + a_{2t} - \theta_{221} a_{2t-1} \quad (3.5.6)$$

The assumption made before implies that a_{1t} is independent of a_{2t}, so that the matrix γ_a and the θ_i matrices are diagonal

$$E [a_{1t} a_{2t}] = 0 \quad (3.5.5)$$

where ϕ_{111}, ϕ_{121}, ϕ_{211} and ϕ_{221} are the elements of the autoregressive parameter matrix Φ_1. Note that the first two subscripts of the parameters designate the row and column of the matrix; the last subscript denotes the subscript of the parameter matrix Φ_1. The same is true for θ_{111}, θ_{221}, and the moving average parameter matrix θ_1.

(*i*) *Time Delay*

Finding the dead time or initial delay is one of the crucial steps in transfer function or control analysis. The method of finding this lag consists of using parameter estimates and their plots.

The delay between two different series can be obtained by fitting a high order ARV(n) model using the inverse function coefficients. The method use to calculate the inverse function coefficients and subsequently ϕ and θ for EARMA model is a generalization of the univariate ARMA case [sec 3.3.2]. The formulation of the initial values of extended ARMA models is

discussed in [4].

After the inital values have been calculated using the inverse function coefficients, the confidence interval of the parameters is studied. The number of $\phi_{ij.\ell}$ terms that are effectively zero (which include zero in their confidence limits) prior to a significant parameter indicates the delay. This is so because the $\phi_{ij,\ell}$ constitute the impulse response function. Once it is seen that the estimate of the initial delay or dead time does not change even after increasing the order of the model, one can stop, choose the value of delay, and compute the parameter estimates. Any discrepancies or uncertainty in the determination of lag L can be corrected in the final estimation and checked by the F-test together with the other small parameters whose confidence intervals include zero.

Plotting the parameter estimates of the auto-regressive model gives a much clearer and stable picture of dead time.

For an ARMA(n,m) model, it usually suffices to compute m + n coefficients. but when an initial delay or dead time L is present, the first L coefficients are theoretically zero and it is necessary to compute at least L+m+n coefficients. Thus if the lags are large,

one may pay a price for this in terms of extra computation. [26]

(iii) Diagnostic Checking

All the diagnostic checks described in the ARMA model building strategy are applicable for EARMA models with the addition of one more test. The need for this test arises from the basic assumption made in formulating EARMA models from general ARMAV models; i.e. the independence of input noise with output noise. This assumption is validated by checking that the plot of cross correlations between input and output residuals lies within the standard error limits of $2/\sqrt{N}$. The residual series a_{it} and a_{jt} are first average subtracted. Then the auto and cross correlations are computed by

$$\hat{\gamma}_{aijk} = \frac{1}{N} \sum_{t=k+1}^{N} a_{it}\, a_{ij-k}\,, \quad \hat{\gamma}_{aij(-k)} = \hat{\gamma}_{ajik} \quad k \neq 0 \tag{3.5.7}$$

$$\hat{\rho}_{ijk} = \frac{\hat{\gamma}_{aijk}}{\sqrt{\hat{\gamma}_{aiio}\hat{\gamma}_{ijo}}} \quad , \quad k = 0,\ 1,\ 2,\ldots.$$

(iii) Extended ARMAV Modeling Algorithm

The modeling procedure for the extended ARMAV models is similar to the univariate case.

Step 1

Obtain the initial values of the parameters by the inverse function method [4]. The initial j values also provide an indication of the presence of initial delay. However, it should be noted that the choice of the initial delay L is tentative, and if there is doubt, the parameters should be included in the final estimation.

Step 2

By the nonlinear least squares method, fit the models for input and output separately, in a form modified for lag, if necessary, starting from the initial values in step 1. Increase the value of n until the reduction in the sum of squares is insignificant.

Step 3

Drop the small parameter values to reduce the order and/or increase the lag or dead time and refit the model. Check the refitted model against the one in step 2 and determine the adequate model.

Step 4

Compute the auto correlation of residuals of all series and the cross-correlation between different a_t's using (3.5.6) If the correlation lie within the $2/\sqrt{N}$ band. The residual series are independent and hence are adequate model has been obtained. If not repeat steps 2 and 3 with a greater number of parameters.

Chapter 4

DEVELOPMENT OF THE PACKAGE

4.1 Selection of Hardware and Software System

Selection of hardware and software system for the package was considered to be of critical importance. Three general criteria were considered for system selection.

(1) Local availability of the system software and support.

(2) The operating system of the computer should be powerful and widely used so that a wide range of software is available.

(3) The cost inclusive of hardware and software, should be kept at a moderately low limit. In this case, the budget was limited to SR.50,000/-

Following an extensive examination of current microcomputer applications, in university, industry and local business, the following system was chosen.

HARDWARE

Computer System: An IBM Personal Computer (PC) which is now considered as the *defacto* standard of micro computer industry was chosen. The IBM PC enjoys a clear lead over

other systems because of the wealth of hardware and software support available. In view of anticipated high storage requirements, an IBM Personal Computer XT was selected, as it offers convenience, greater storage capacity, faster access to programs and broad expansion and information handling capabilities.

DISPLAY SYSTEM:

The interactive nature of the application software required color graphics combined with a clear text display. The dual mode display system of the IBM-PC makes it possible to install the IBM color display and IBM Monochrome Display on the XT. This offers the advantage of displaying text on a green screen while data are plotted simultaneously on the high resolution color graphics monitor.

Printer and Plotters

For data logging, print-screen and compilation purposes, an IBM Graphics Printer was selected. It is a versatile, low-cost, quality matrix printer that prints both graphic images and an extensive range of text characters at a moderate speed of 80 characters per second. However, graphic image output from a dot matrix printer is low resolution and monochrome. To fully capture color graphics screen displays, a Hewlett-Packard 7475A graphics plotter was chosen. This is supported by almost all the software packages for the PC. Moreover, it

has a 6-pen carousel and ability to plot on different paper sizes and transparencies.

Execution Speed and Memory Size:

Low execution speed and limited memory size were considered to be the two main handicaps of microcomputers in general, although they can be overcome to a great extent using the IBM-PC. The Math Co-Processor option was chosen for the IBM-PC which provides an 8087 companion processor to the main 8088 chip. The highlights of the 8087 performance are [35]L

- It multiplies 32-bit and 64-bit floating point numbers approximately 80 times faster than the 8088.
- Its 80-bit bus provides 5 times the precision of the 8088's 16-bit bus.

Although the 8087 chip was considered essential for implementation of complex numerical algorithms, the resulting object code from the algorithms was not expected to run on the basic RAM for the XT. The 16-bit IBM micro provides the facility of directly addressing up to 640K of RAM memory, but on the system mother board there is space for only 256K of memory. A deficiency of the IBM PC is that it has only four expansion slots, as compared to eight on Apple II for example, but this deficiency was the source of major

innovation in microcomputer hardware: that of multi-function cards. The Quadboard multi-function card (Quadram Corporation) was installed on one of the expansion slots of the XT. This is supplied with 256K RAM, so that the memory of the machine was enhanced to a sizable 512K.

The multi-function board provided useful RAM disk software. A RAM disk is a portion of the computer's RAM to which a program and data can be loaded and used as if it were a disk drive. Because a RAM disk has no moving parts, it performs disk operations nearly 50 times faster than a floppy disk drive and about 10 times faster than a typical hard disk [36]. Similar to a RAM disk is the useful printer spooler on the Quadboard, which means that the PC is rarely tied up with printer tasks. The real time clock on the board relieves the user from typing the current date and time whenever the system is booted and it also maintains valuable chronological records of the created directory and files.

The IBM personal computer can support two serial and two parallel printers at the same time. The IBM-XT is supplied with one RS-232C serial card and one parallel interface was made available with the IBM monochrome graphics adapter. Two more interfaces, one serial and one parallel, were supplied with the Quad board. This increased interfacing capability greatly enhances the power of the machine. One parallel port was utilised for printer connection while the plotter was

connected through the serial port. The second serial port was reserved for interfacing with a process computer; either through regular telephone lines or using a coaxial cable.

SOFTWARE

Operating System:

The central processing unit (CPU) or microprocessor of microcomputer is the core of computer's hardware, but the operating system serves as the core of the computer's software. The standard operating system for the 8088 chip is based 16-bit IBM is the Personal Computer Disk Operating System (PC-DOS). IBM PC-DOS V2.10 was chosen for this work. As well as giving software support, DOS offers a friendly image with powerful, yet easy to use functions. Three of the most important characteristics of DOS V2.10 are:

(1) Ability to adapt to different advanced hardware devices, such are hard disks, local area networks and plotters.

(2) Expertise in automatically supporting programs that cross the 64K mark of the 8086/8088 microprocessor.

(3) Provision of features such as a hierarchical directory system, input/output redirection and piping, making facilities of the UNIX operating system available to microcomputer users.

Programming Language:

DOS 2.10 supports many different programming languages most of which are subsets of their mainframe versions. In order to implement complex mathematical and statistical algorithms, different options were studied.

The IBM machine has a built-in BASIC language interpreter. Although the interpreter has extensive capabilities of graphics, screen control and communications, it is slow and its code cannot exceed the 64K limit. Compiled basic was considered as an alternative. Although this does not suffer from the above mentioned problems, it does not support graphics. Furthermore, the lack of any standard for BASIC language means tht BASIC programs are not easily transferable and few engineering packages have been developed so far in BASIC in this area of application.

Other major languages available on the IBM PC including PASCAL, APL, and LISP etc. were rejected due either to reservations about their first time implementation on microcomputers or their non-mathematical nature.

Consequently FORTRAN was chosen. The vast majority of scientific and engineering software available on mainframe computers has been written in FORTRAN and FORTRAN 77 at a subset level has long been implemented on micro computers.

The Microsoft FORTRAN 77 compiler (Ver 3.20) was chosen for programming work. The MS-FORTRAN compiler includes a number of extensions that provide feature of the full FORTRAN 77 [36]. This means that FORTRAN programs on a main frame can be transported to the IBM-PC with little changes. The compiler fully supports the 8087 coprocessor giving the FORTRAN user much faster program execution and accuracy. The language also links with assembly language and MS-Pascal programs, and powerful input-output facilities give the Ver 3.20 user an interactive environment similar to BASIC.

Screen Editor: Using a high level language such as FORTRAN demands a good editor which has the ability to screen scroll, move copy blocks of data and has search/replace commands. The EDLIN editor supported with DOS is only a line editor, and is of little practical value: Instead, the IBM professional editor, V1.0 was acquired, as it offers the above mentioned functions with merge file, tab-setting and macro entry facilities.

Integrated Software:

The FORTRAN compiler provides a suitable programming tool for complex mathematical tasks, but the requirements for graphics and communication ability still remain. It is essential to provide an environment from which the user can control and call any desired task or functions.

With the advent of microcomputers, software manufacturers were quick to realize their potential in calculations, word processing, data base management, graphics, telecommunications etc. Although adequate packages were available for all of these basic applications, soon many problems arose. If data generated from one program was to be used for another application intermediary files would have to be created. Time consuming techniques were required to import-export and translate these files. The task of acquiring several packages and studying each of them, so as to avoid mixing commands was both expensive, unnecessarily difficult and time consuming. Moreover, no integration of the facilities was achieved.

Integrated software packages offer a solution to the problem i.e. how to combine the basic applications into an integrated whole. A unified package eliminates intermediary files and presents a consistent common structure to perform any analysis or create any type of function or report. The microcomputer would become a work station and the integrated package a tool box containing all the facilities that most users are likely to need, regardless of the application [37].

The present requirement is for an "integrated" package that includes a spread sheet, word processor, data manager, graphics generator and communication services. Two packages, Symphony and Framework, offer these advanced features.

Symphony was adopted as it has superior capabilities in spread sheets, communications and integration.

In addition to the five basic applications, Symphony provides a windowing facility by which different functions can be viewed at the same time. Its powerful macro language facilitates sophisticated applications using its unique menu system. It has a self-learn mode, remembering keys as they are typed, and its DOS application command gives access to a multitude of software. Six different types of graphs can be displayed and plotted by symphony using different colors, fonts and sizes. Finally, the availability of an on-line relational "Help" facility and ability to capture data directly into the work sheet from any linked computer or device makes Symphony an ideal choice for this application.

4.2 Interactive Implementation of Box-Jenkins Technique

Box and Jenkins envisaged how their modeling procedure would be programmed [6].

> "The computer has great speed and precision, compared with which the human mind is slow and inaccurate. However, the mind has powers of critical and inductive reasoning, relative to which, those of the present day computer are minuscule. This makes possible a happy division of

labour in which mind and machine each contributes what it does better. Models built entirely automatically by the computer without the intervention of intellectual imagination and restraint are unlikely to be very good, and could be very bad. Because of that interactive programs are called for, in programming the technique. When properly used, these programs would allow the investigator great freedom to experiment with his data, and can spark the ideas necessary to creative iteration".

In reviewing the time series software in chapter-1 a number of shortcomings were observed:

(1) Firstly the above idea conceived by Box and Jenkins, in which the analyst has full freedom and flexibility to make his decision using the various visual aids, was rarely to be found. A few interactive packages on main frames offered such capability [26,27] but only at the expense of high cost and non-portability.

(2) Nearly all programs were rigid, so that although aimed at wide range of users, customizing these time series packages for certain specific application or enhancing them with new routines or

algorithms was difficult.

(3) None of the time series packages provided a process interface. They were designed for analysis of past data, for forecasting and report purposes, and no attempt was made to use the identified model on-line.

Bearing in mind the above details, design was undertaken for a portable, user friendly, adaptable and on-line time series package.

The first phase of the project involved the development of stand along FORTRAN subroutines, each subroutine performing a specific task. The use of subroutine modules facilitates adaptation to different problems. Extensive use was made of MS-FORTRAN file-handling and input-output instructions. These facilitated the creation of interactive FORTRAN programs and the transfer of information between different program modules. An attempt was made to generate the fastest and most economical object code by using the 8087 link library and the special metacommands of the MS-FORTRAN compiler.

Each of three stages of the Box-Jenkins technique, model identification, estimation and diagnostic checking were subdivided. For example, the identification stage

involves the data plotting, calculating and plotting of the auto and partial autocorrelation test for stationarity and the need for differencing transformations etc. The sub-divisions were made at stages where the analyst is required to play a major role. All subdivisions were programmed and saved as separate stand-alone object modules.

With the subprograms completed, the whole Box-Jenkins procedure was available but in a fragmented form, and it was then necessary to integrate the parts to create a sound application program.

Application programs are usually executed under a menu driven system because of their convenience and intuitive appeal . The purpose of the menu is to limit the choice of the action in such a way so that the list of possible actions can be presented to the user, who decides on one of them. In the present application, several menus are required. There are different solutions on how to work with several menus. If one statement from the list of a menu is assigned to another menu, a vertical menu structure is achieved. This makes it possible to define not only a tree of menus, but also any connected graph of menus. Another possible shape is the linear horizontal structure of menus which facilitates movement between the predecessor and

successor of a current menu using predefined keys [38].

One rudimentary way ot implement menus is to write a large FORTRAN main program which will control the menu structure and link up all of the object modules into one execution module. Numerous screen menu writing supporting routines would also have to be included. However Microsoft's elementary screen control functions and slow speed of screen display, coupled with the expected large size of the execution module, make this an inefficient system. Furthermore, since FORTRAN does not directly support graphics, this would only be a partial solution.

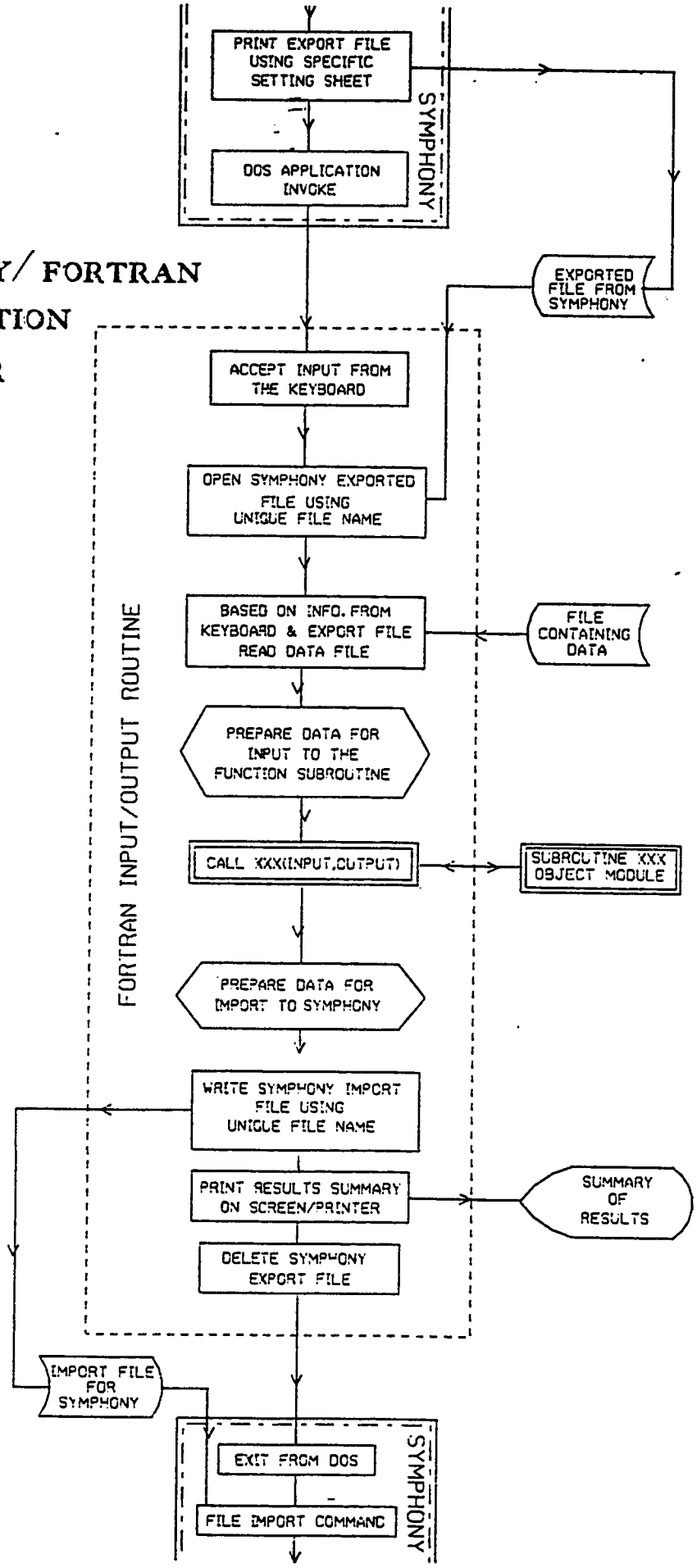
In contrast, Symphony's powerful macro language allows the user to program a complex menu driven system with ease and with a minimum number of commands. Menu calls can be made to another menu, subroutine or location and any mistakes can be rectified by returning to the predecessor menu using the ESC key. Although Symphony' work sheet and windows fill most of the screen, the upper two lines are reserved for the command menu. This comprises a list of command words and one is selected either by moving the pointer and pressing RETURN, or by typing the first letter of the command word.

The next stage of the design was to integrate the FORTRAN subroutines into the Symphony environment so that smooth information transfer can take place. Files are the most common means of data storage, but not all files were stored using the same format. FORTRAN programs can easily read and write ASCII files; Symphony was found to capture ASCII files using the file Import command, and after some experimentation it became clear that Symphony exported ASCII files using the PRINT USING setting sheeting command. Thus a common channel for data transfer between fast mathematical FORTRAN subroutines and the Symphony integrated package was established. Need was felt for a main program for every subroutine module to be known as the input/output routine. The architecture for these routines is shown in Fig.4.1. An input/output routine has the function of providing data format compatibility between a FORTRAN subroutine and Symphony.

An input/output routine receives information from a unique file exported from Symphony or from keyboard input. Based on this information the routine reads files containing specific data and then transfers control to the subroutine to perform a certain statistical function. A summary of results from the subroutine is displayed and a hardcopy can be obtained using the Prt

SYMPHONY/ FORTRAN INFORMATION TRANSFER SYSTEM

Fig. 4.1



Scr command. The I/O routine then writes the output into a unique filename using special formatting for data titles and labels so that the Symphony work sheet can incorporate it. Finally the I/O routine deletes the exported file from Symphony so that the old data is not confused with the data directed to it by Symphony on the next iteration.

Graphic displays are an essential part of the B-J technique, as most results are graphs and most decisions are based on inspection of the graphs. Having designed a hierarchical menu and a file structure to integrate data and application files , the remaining task was to incorporate graphics capabilities. Symphony offers six different graph types in vivid colors. High-low graphs provided by Symphony were ideal for displaying maximum and minimum data values. For correlations, line graphs provided four different colors and provision for +/- limits. To observe multiple graphs simultaneously, the windowing feature of Symphony was used, giving the user flexibility to zoom and magnify any window or any portion of a graph. The use of setting sheet, attaching a graph to a window, and good labeling procedures enhance the power of Symphony Graphics. A graphics system was designed in which invoking a DOS application by a command menu leads to execution of a specific

FORTRAN program. After execution the system returns to Symphony, which imports the newly-created data or file into the worksheet and transfers control to display the results. Sometimes two or more graphs should be displayed simultaneously. This was achieved using the Symphony windowing capabilities. A facility was provided to save the graphic screen for subsequent printing.

Each step of the Box-Jenkins procedure was implemented in the form of Symphony command menu using the Symphony macro language. Box-Jenkins is an integrated fully self-checking set of procedures in which all initial assumptions are subsequently verified or changed. Each menu step in this application gives the analyst time to make and verify his decisions. Symphony's word processing facilities were also used when it was felt necessary to provide relevant information to supplement visual displays. If the user is not satisfied with the plots, he returns to the FORTRAN application routine with new parameters and analyses how possible alternatives can be accommodated.

The question of the process interface is covered later in this chapter.

4.3 Implementation of Pandit-Wu technique as a Stand Alone Program

The Pandit and Wu method is an automatic batch time-series modeling technique. This algorithm minimizes user interaction and reliance on correlation graphs. Statistical procedures are used in which successive ARMA (n,n-1) models are fitted for n = 1,2,3,... until model adequacy is achieved [31].

In this case a large stand-alone program (see section 1.3) with its associated subroutines is appropriate. The complexity and size of the algorithm requires a high level language, such as FORTRAN and the Microsoft FORTRAN V3.20 was again used. However the method is computationally intensive and its implementation on a microcomputer was not expected to be efficient. Nevertheless many of the advantages of the microcomputer previously cited could also be demonstrated for the implementation of the Pandit and Wu technique. A system of files was devised which is shown in Fig.4.2 and discussed in the following paragraphs.

Stand-alone programs accept input in a rigid fixed format which is both tedious and confusing, as data entry is mostly carried out in an input file. To ensure that all entries and columns are in order the user must consult an instruction manual each time he executes the

MICROCOMPUTER IMPLEMENTATION PANDIT & WU TECHNIQUE

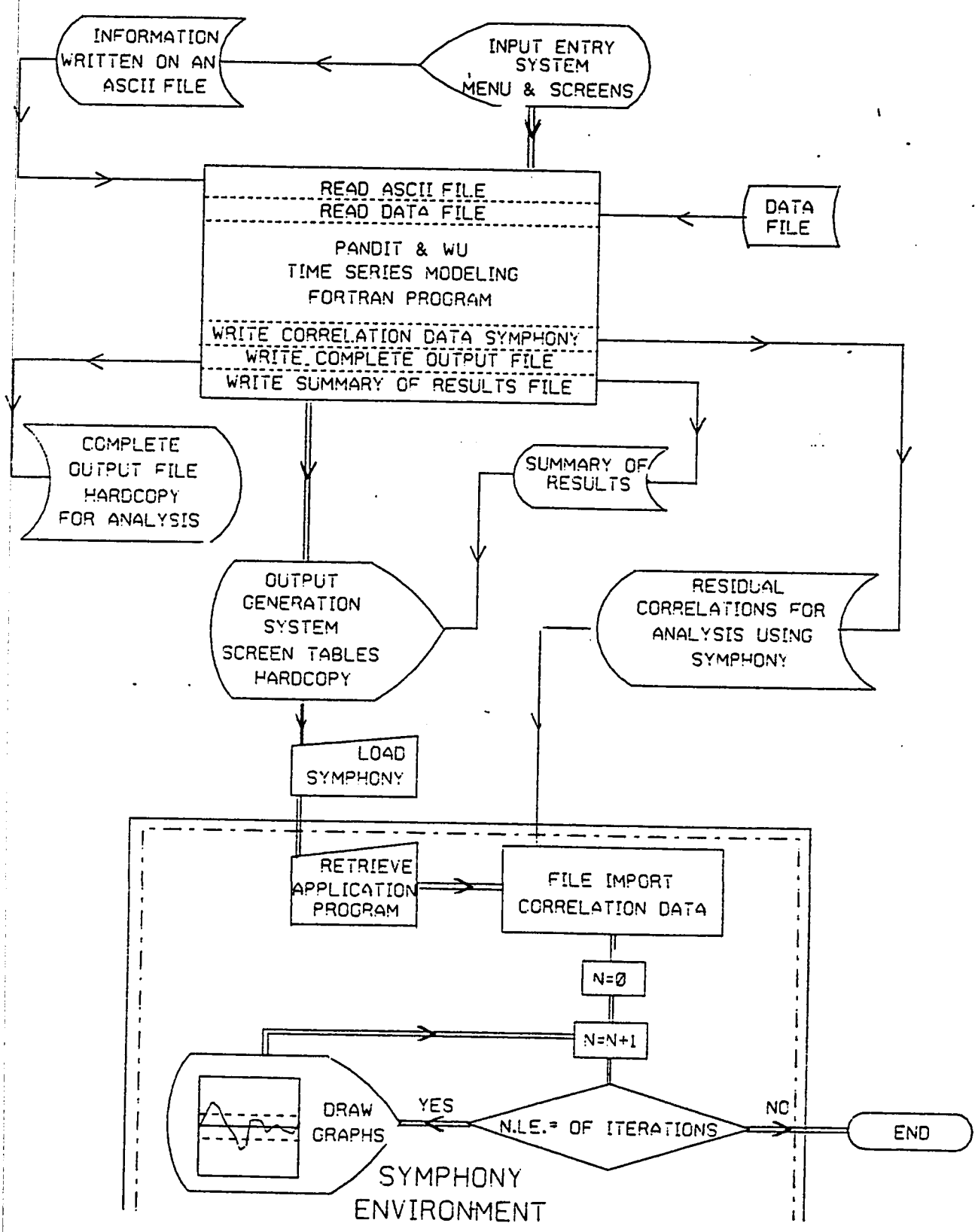


Fig. 4.2

program. Consequently Microsoft FORTRAN was considered inadequate, but potential was seen in the IBM interpreted BASIC.

The DOS BASIC supports many useful screen functions such as underlining, video intensification, blinking and reverse video. An input entry system was written in interpreted BASIC and this was then linked with the main FORTRAN application module using a DOS batch file. The input entry system consists of series of screen images. Each image consists of a familiar border, details and description, about the input field, and a reverse video input field for data entry. All screen images were saved in binary form to permit fast access. The system first collects all information needed by the FORTRAN program using convenient and friendly displays and then writes them in a customized way on an ASCII file so that FORTRAN program can interpret it as an input file.

After execution, the stand alone programs dump a large amount of output as a series of print-outs on files. The user must extract useful or relevant information from the stack of listings. To overcome this inconvenience the BASIC interpreter was again used. Although the large listing from the Wu - Pandit analysis was retained, a secondary file was created onto which the most important results such as model order,

parameter, and confidence estimates, residual sum of sequences (RSS) and F-test etc. were written. Following execution of the main Pandit and Wu program the DOS Batch file transfers control to the BASIC output generating system, which reads the most relevant results and then displays them in a convenient tabular fashion. If more than one series are involved, more tables will be created, and the analyst can switch between the tables and produce a hard copy of the summaries using predefined function keys.

Although Pandit and Wu emphasize independence from graphic displays, in the final stage of diagnostic checking, they check the model adequacy by plotting auto and cross correlations of residuals of different series [4]. These are generated in the extensive output file, but to link them to Symphony another file was created. After execution of the job, Symphony is booted. A small menu-driven application program was written. The application menu first offers a list of available files generated for Symphony by the Pandit-Wu programs. The selected file is loaded into a specific location in the worksheet. Then depending upon the number of series and number of iterations, the correlation graphs are displayed one iteration after another. In the case of multiple series, all correlation graphs were windowed

and arranged so that all graphs resulting from a single iteration are displayed at the same time on screen for convenient comparison. An option is available for enlarging or zooming and image saving any of the displayed windows.

4.4 Process Interface

Development of Box and Jenkins, and Pandit - Wu techniques on a microcomputer has provided powerful and sophisticated tools for forecasting and control of dynamic processes. To fully exploit the potential of those application programs and to provide wider access to the package, it was thought essential to establish a means of interaction between the programs and the process itself.

The present application is oriented towards process industries where process computers are responsible for the important task of smooth control and operation of the plant. The process computer has information handling and data processing capabilities just like an ordinary computer, with a ready interface to the whole process via a series of transducers and analogue and digital devices. To access this information about the process it is necessary to read the records stored by the process

computer. However, since process computers have the critical task of plant regulation, this procedure should be established so that normal operation of the process control computer is not placed at risk.

The Symphony integrated package, which was already used for integration of FORTRAN routines, graphics, word processing and command menus, was a logical choice for inclusion of communication ability into the package. Symphony offers a communication environment in which data can be received and transmitted to other computers, via modems and regular telephone lines. Unlike other communication programs, Symphony captures data in the work sheet as it is received. This can then be processed using Symphony's spread sheet word processing, graphics, or data base management capabilities. Furthermore, it is possible to temporarily interrupt the communications session, analyze data and then immediately return to the communications session. The limitation found in Symphony's communication capability was that it can only communicate through telephone lines, so a dedicated telephone link between the microcomputer and process computer must be established. Also a process computer terminal would be tied up in data transfer with IBM PC.

A closer hardware study of process control computers revealed that most of these computers are customized versions of popular minicomputer such as DEC PDP/II, VAX-11 and Data General Nova etc. In other cases it was established that the terminals use one of the many standard terminal protocols such as DEC VT100, ADDS Regent etc. The popularity of the IBM PC has led to the development of terminal emulation programs for the microcomputer.

Thus a serial interface can be established between a process computer terminal and the IBM PC. Process computer software supports many peripheral devices and no modification is necessary to the process computer software other than to specify the existence of an additional peripheral. The PC is then able to issue standard process computer user commands and receive replies from the process computer.

Symphony can be used to achieve full system integration. In order to accommodate the terminal emulator program, the DOS Application Invoke command of Symphony is used. Smart communication and emulator programs are available in which the PC can communicate completely unattended and also data transfer may even take place when another program is under execution [39]. To formalize the process of data collection an

application routine may be written which after each iteration of the Box-Jenkins technique, invokes the emulator program so that data transfer from process computer can take place. The received data may be incorporated into a Symphony work sheet and catalogued using Symphony's date and time function. Symphony's data base management facility then updates the file and orderly records of relevant data are kept. After this pre-processing of data the Box-Jenkins technique can proceed.

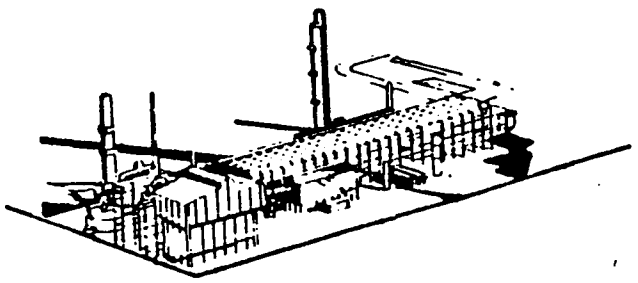
4.5 Final Perspective

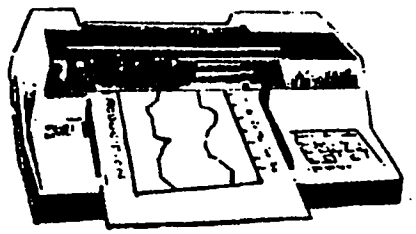
The work described here included development of efficient FORTRAN routines, common menus, essential graphics, process interface and the final integration and implementation of all function using Symphony has provided a sophisticated time series analysis system, which can be readily applied to different processes and experiments. Fig.4.3 on the next page clearly defines the place for this system in the process control room.

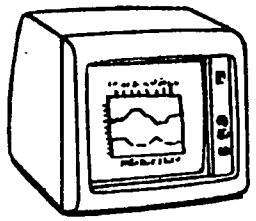
A typical session starts with a terminal-emulating call from the IBM-PC to the host process control computer to get updated file of process data. The newly acquired file is loaded into a Symphony work sheet for

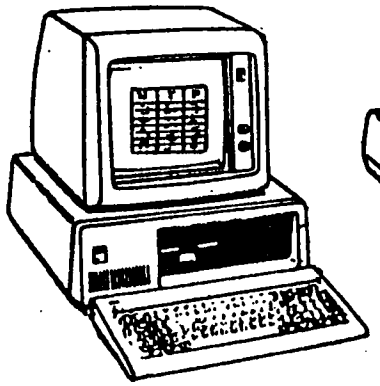
A PLANT DIAGNOSTIC TOOL

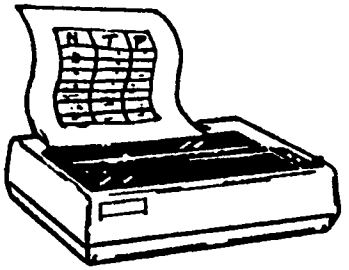
Fig. 4.3











pre-processing. The Box-Jenkins application program is then started which, after series of well-defined steps will give the analyst a parsimonious transfer function model of the process data. This gives accurate accurate information about the process gain and dynamics. Using the forecasting program he can then predict the forthcoming trends in the plant variable, and appropriate action in the form of controller tuning and adjustment can be readily made on the process computer. This can be repeated some time later to see how immediate feedback between the machine and process would results in marked reduction of plant disturbances and more accurate forecast of the process can be obtained. The automatic technique of Pandit and Wu can also be readily applied, as an alternative to Box-Jenkins with the possibility of obtaining with minimum user interaction.

The relevance of this system to the control room implies that the control engineer would be the most probable user of the package. He can use the package in his own time, to optimize existing or developing new production plans. The availability of vast graphics software and hardware enables him to create superior reports and charts for presentation to the plant management.

Chapter 5

RESULTS

5.1 Gas Furnace Data

Transfer function modeling has been carried out using real data to test the effectiveness of the techniques used and to validate the operation of the package. The analysis was performed using both the Box and Jenkins, and Pandit and Wu techniques and their relative merits were compared.

The real data used is from the "Series J" provided by Box and Jenkins[Appendix C]. The data was obtained from a gas furnace and consists of 296 successive pairs of observations (X_t, Y_t) extracted from the continuous records at 9 second intervals. The input (X_t) is the coded input gas rate into the furnace. The resultant output concentration of carbon dioxide leaving the furnace is taken as the output (Y_t).

5.1.1 Transfer Function Modeling Using Box & Jenkins:

As a first step in the transfer function analysis gas furance data were plotted to observe any prevailing trend, seasonality and non-stationarity. A plot of the input time series (Fig-5.1) shows that no non-stationary elements are apparent and hence no transformations for series 1 are required.

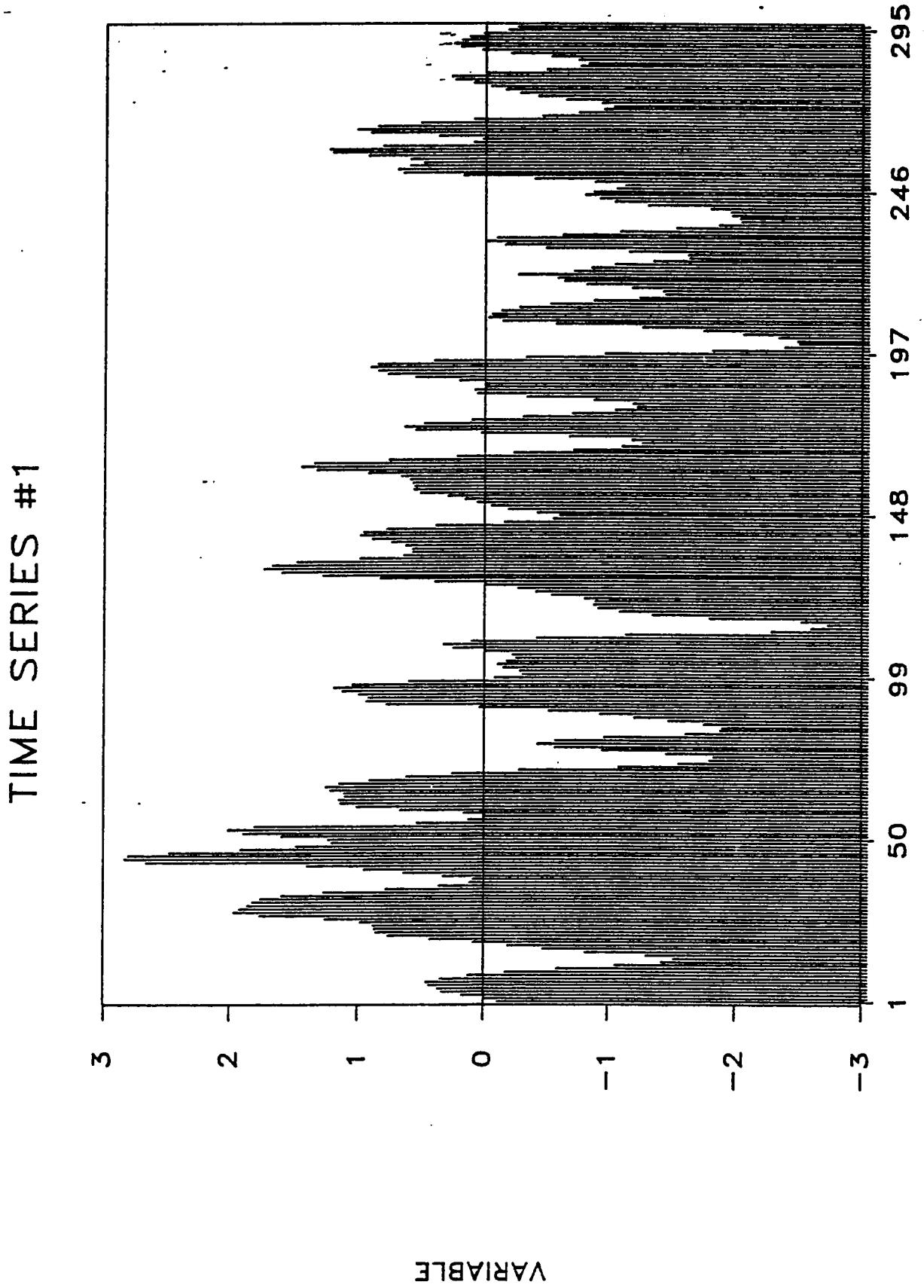


Fig. 5.1

The building of an ARMA model for the input series begins by calculating the autocorrelations and partial autocorrelation function for the input series. The plots of these estimated correlations (Fig-5.2) damped out fairly quickly again confirming that no differencing was necessary. A tentative model order for the input series is deduced from (Fig-5.2). The autocorrelations are dominated by a mixture of exponentials or damped sine waves, so no moving average part was suspected. Furthermore only the first three partial correlations are non-zero. This means that the input series may be described by a third order autoregressive process or ARMA(3,0) model.

One test suggested in [20], to aid in choosing an ARMA model from correlation graphs is to take the natural logarithms of the series and calculate its respective autocorrelation and partial autocorrelation. Any hidden or ambiguous peaks will show up in these plots. (Fig-5.3) shows the correlation plots for

$$X'_t = \ln (X_t) \qquad (5.1.1)$$

and again it is concluded that an ARMA(3,0) model is appropriate.

Next, the routine to calculate the preliminary estimates of autoregressive and moving average parameters is called,

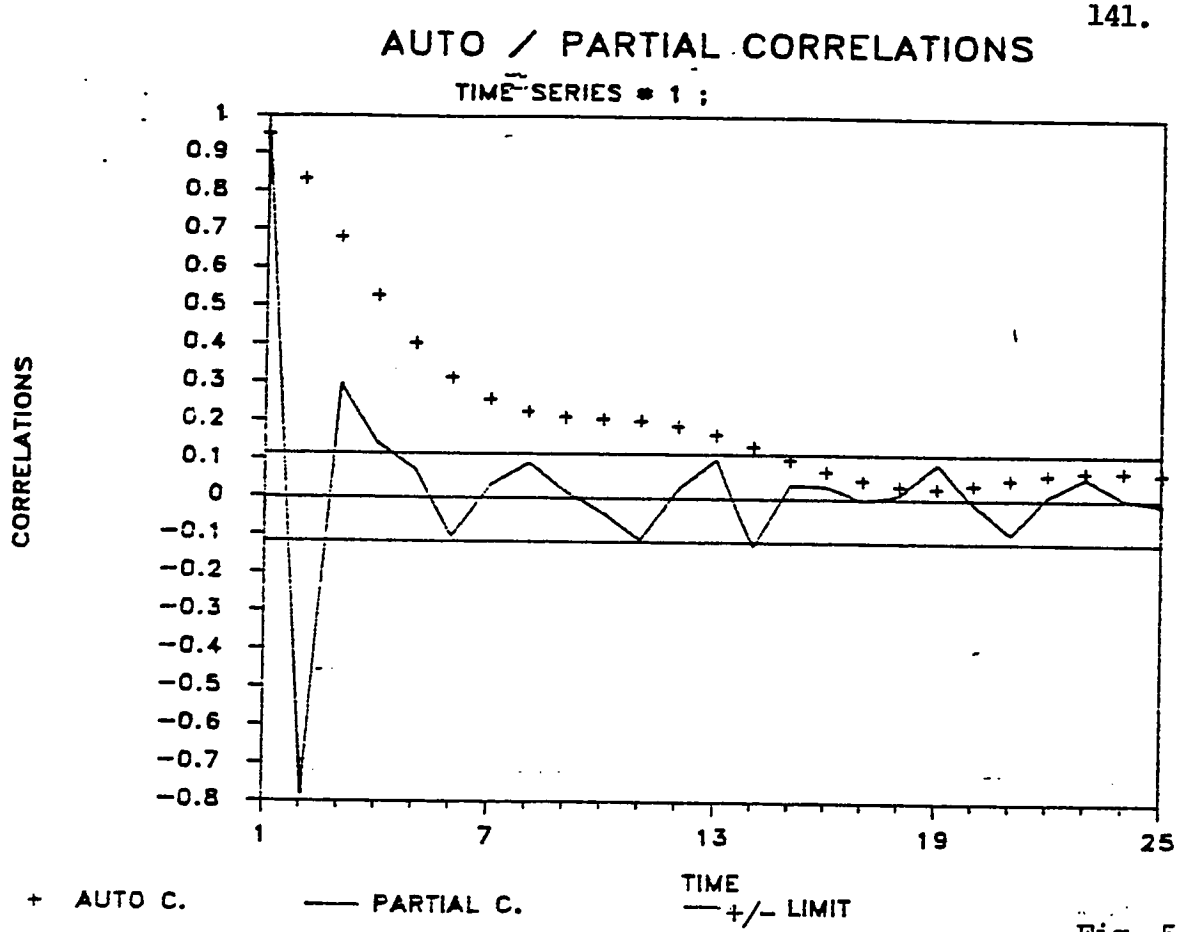

AUTO / PARTIAL CORRELATIONS
TIME SERIES # 1 :
CORRELATIONS
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
1
7
13
19
25
TIME
+ AUTO C.
PARTIAL C.
+/− LIMIT

Fig. 5.2

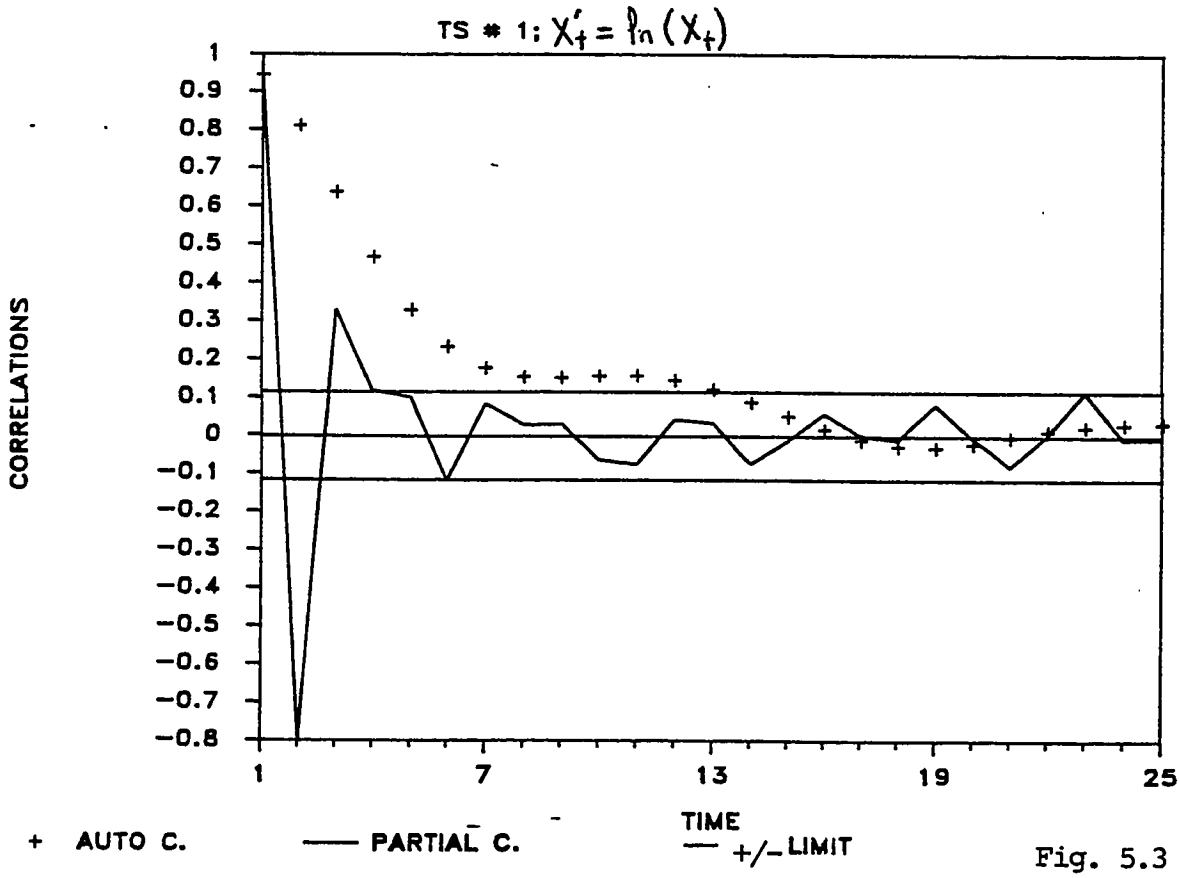

AUTO / PARTIAL CORRELATIONS
TS # 1: $X'_t = \ln(X_t)$
CORRELATIONS
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
1
7
13
19
25
TIME
+ AUTO C.
PARTIAL C.
+/− LIMIT

Fig. 5.3

which gives the following preliminary model

$$x_t = 1.97\, x_{t-1} - 1.36\, x_{t-2} + 0.337\, x_{t-3} + a_t$$

Next, the nonlinear estimation routine is called to minimize the sum of squares of residuals a_t given the above autoregressive parameters as initial values. With the following parameters

Step length parameter = 0.1

Increment factor for numerical derivative = 1×10^{-3}

Convergence criterion = 1×10^{-5}

Convergence was obtained in just two iterations with an elapsed time of less than thirty seconds. The final estimates form the model are

$$(1 - 1.97\,B + 1.36\,B^2 - 0.338\,B^3)\, x_t = a_t \qquad (5.1.2)$$

It is observed that the final estimates are very close to the preliminary estimates suggesting that good preliminary estimates lead to fast convergence of the nonlinear routines.

In order to find the adequacy of this model, autocorrelations corresponding to the residuals of the final estimates are calculated and plotted (Fig-5.4). Observing the

graphs, it is clear that all the autocorrelations lie within the permissible region, and thus the adequacy of the model can be assumed. As a final measure the chi-square statistic test is applied to these autocorrelations to get

$$Q = 29.2$$

The 5 percent critical value of chi-square with 22 degrees of freedom is

$$\chi^2_{0.05,22} = 33.92$$

and there is no evidence to reject the model.

In order to test the stationarity of the output series, first the time series (Fig-5.5) and then its estimated auto- and partial autocorrelations are plotted (Fig-5.6). As with the input series no seasonality or non-stationarity elements are found. As an additional step the series is differenced once and correlation plots (Fig-5.7) are obtained which indicate no major changes in the correlation patterns.

The stationary input and output series are now transformed using the final input model (5.1.2) to yield a prewhitened input series α and a correspondingly transformed output series β with s_α=0.189 and s_β = 0.372.

The transformed series are used to find the cross

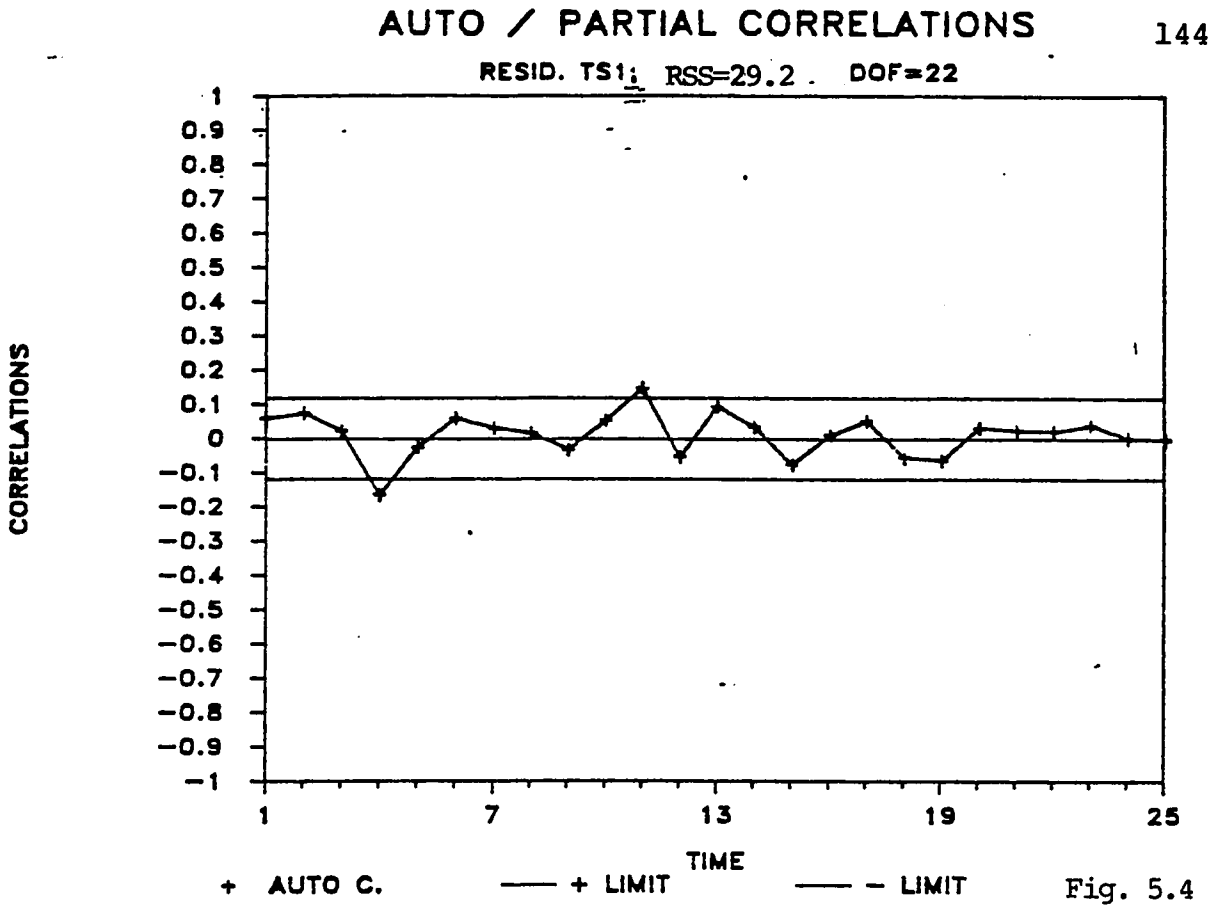

AUTO / PARTIAL CORRELATIONS
RESID. TS1; RSS=29.2 . DOF=22
CORRELATIONS
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
−0.9
−1
1
7
13
19
25
TIME
+ AUTO C.
+ LIMIT
− LIMIT

Fig. 5.4

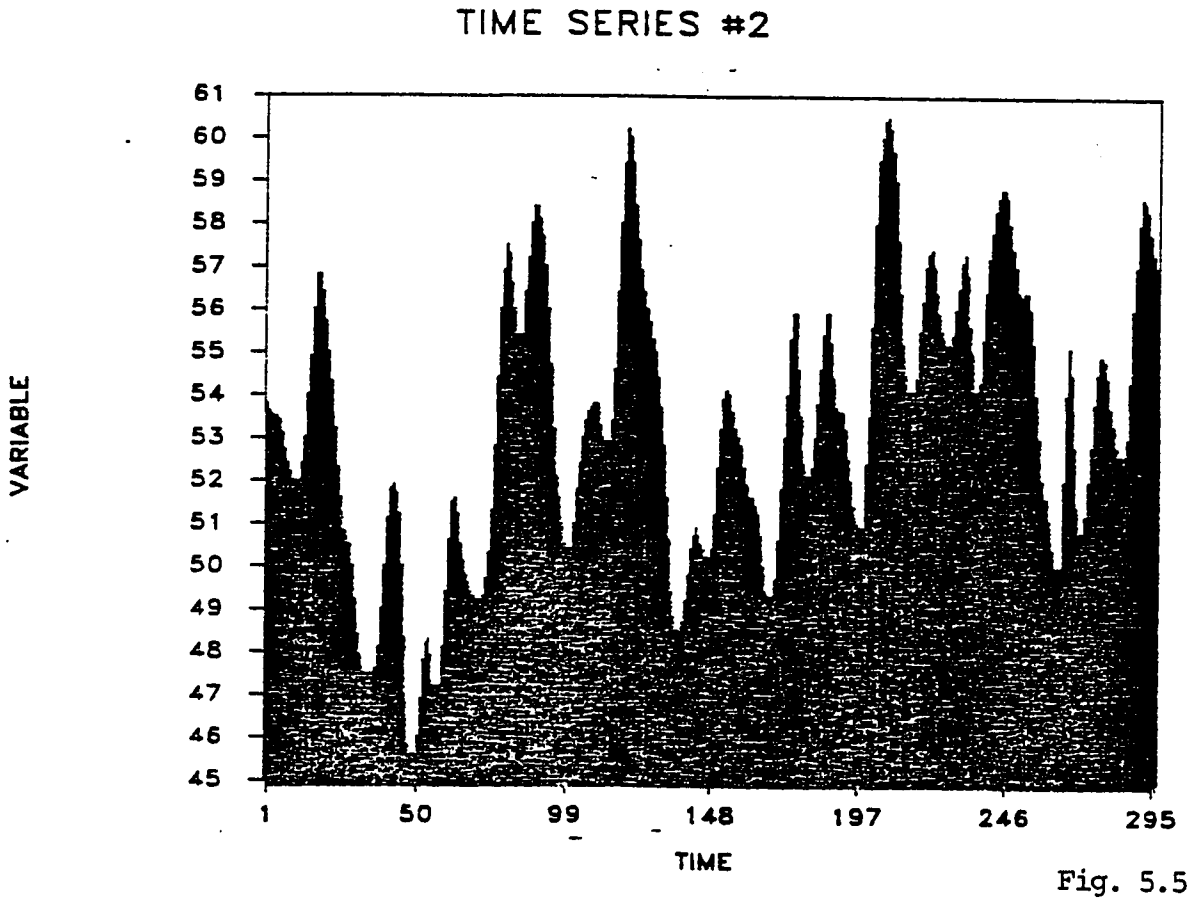

TIME SERIES #2
VARIABLE
61
60
59
58
57
56
55
54
53
52
51
50
49
48
47
46
45
1
50
99
148
197
246
295
TIME

Fig. 5.5

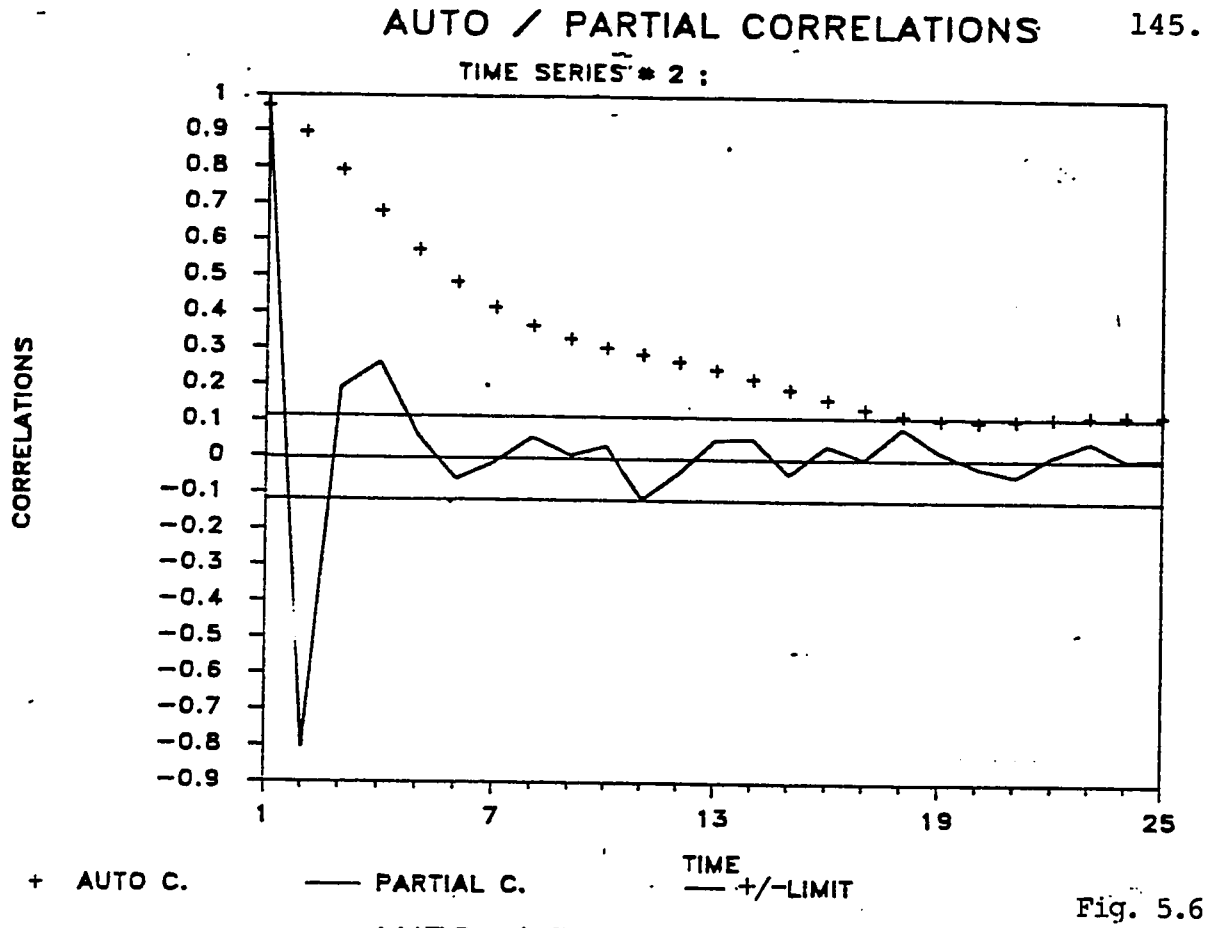

AUTO / PARTIAL CORRELATIONS
TIME SERIES # 2 :
CORRELATIONS
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
-0.8
-0.9
1
7
13
19
25
TIME
+ AUTO C.
PARTIAL C.
+/-LIMIT

Fig. 5.6

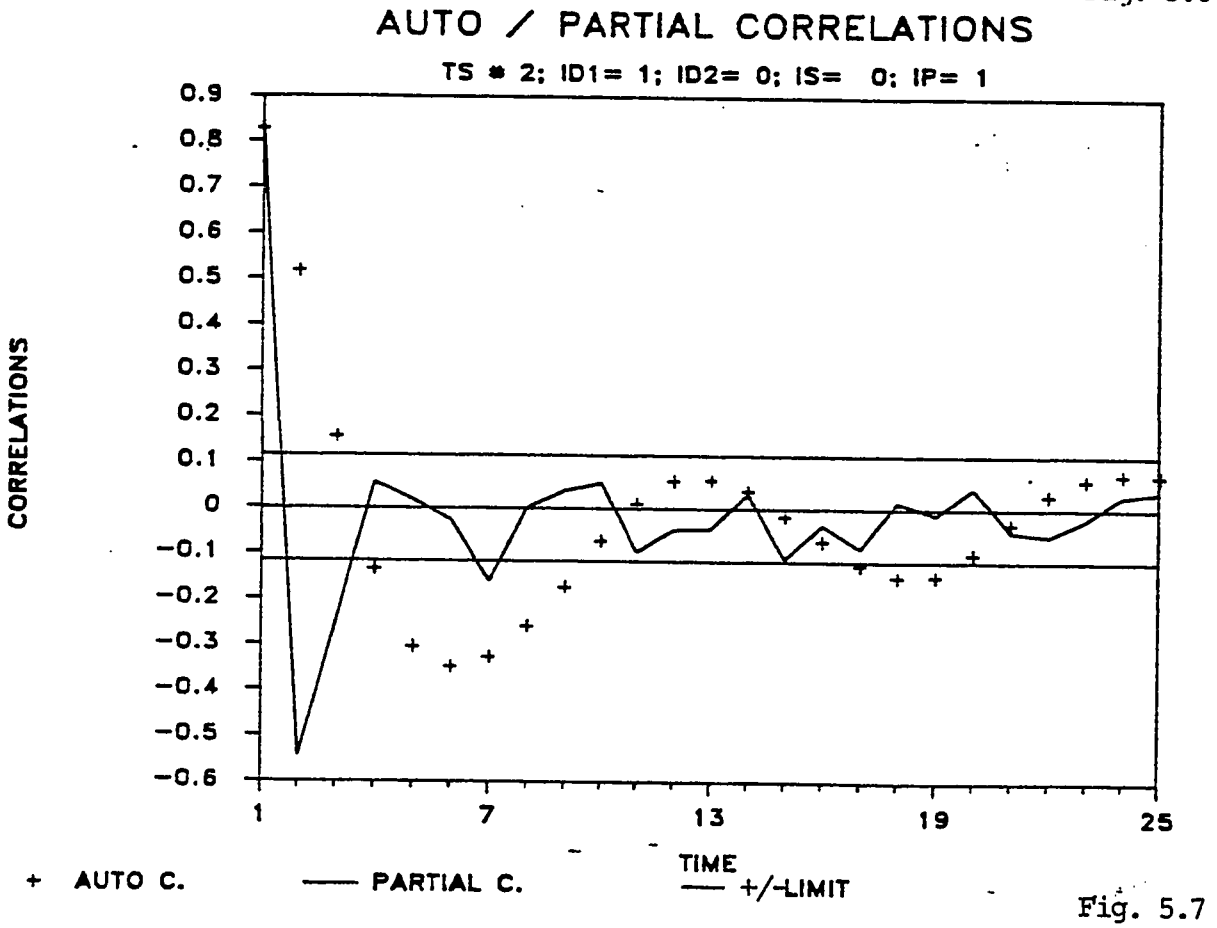

AUTO / PARTIAL CORRELATIONS
TS # 2; ID1= 1; ID2= 0; IS= 0; IP= 1
CORRELATIONS
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
1
7
13
19
25
TIME
+ AUTO C.
PARTIAL C.
+/-LIMIT

Fig. 5.7

correlation function. The estimated cross correlations are calculated and the plot of cross correlation is shown in (Fig-5.8). It is observed that cross correlations up to lag +2 and from lag +8 onwards lie within the two standard error limits and hence they are assumed to be negligible. This suggests that the system dead time is three(b = 3).

To assist in determining the order of left hand side and right hand side parameters of transfer function model, the impulse response and step responses of the corresponding cross correlation function are also plotted (Fig-5.9 and Fig-5.10) respectively. Box and Jenkins have given plots [Appendix B] for sample impulse and step response functions for different transfer function models. Based on the sample plots and guidelines [Appendix B] the tentative model order (r,s,b) = (2,2,3) is selected. Moreover the noise series sequence is generated here using the impulse response weights.

Next the preliminary estimates of the tentative transfer function are calculated to yield the model

$$v_t = \frac{-(0.55 + 0.33\,B + 0.50\,B^2)}{(1 - 0.56B - 0.036\,B^2)}\, X_{t-3}$$

The noise series generated in the last routine is plotted

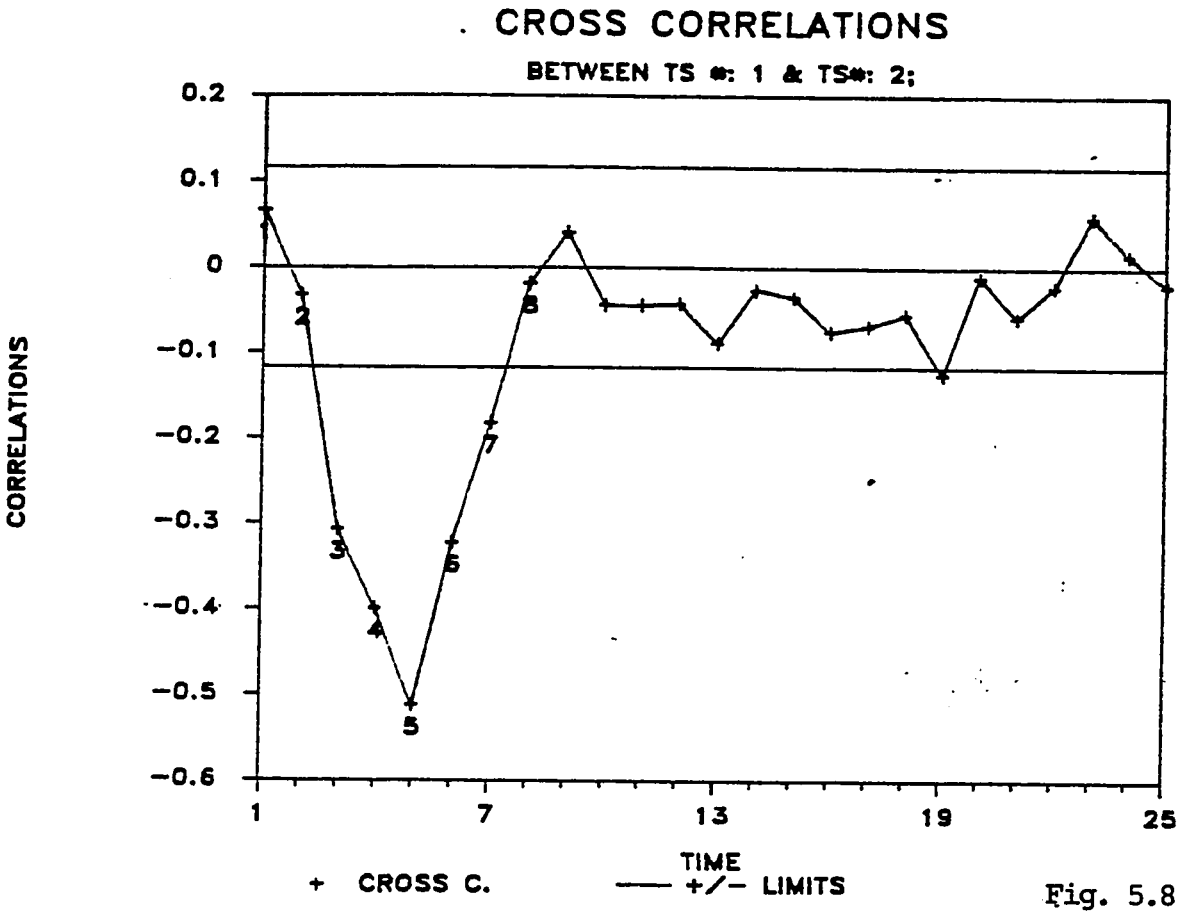

CROSS CORRELATIONS
BETWEEN TS #: 1 & TS#: 2;
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
CORRELATIONS
1
7
13
19
25
TIME
+ CROSS C.
—— +/− LIMITS

Fig. 5.8

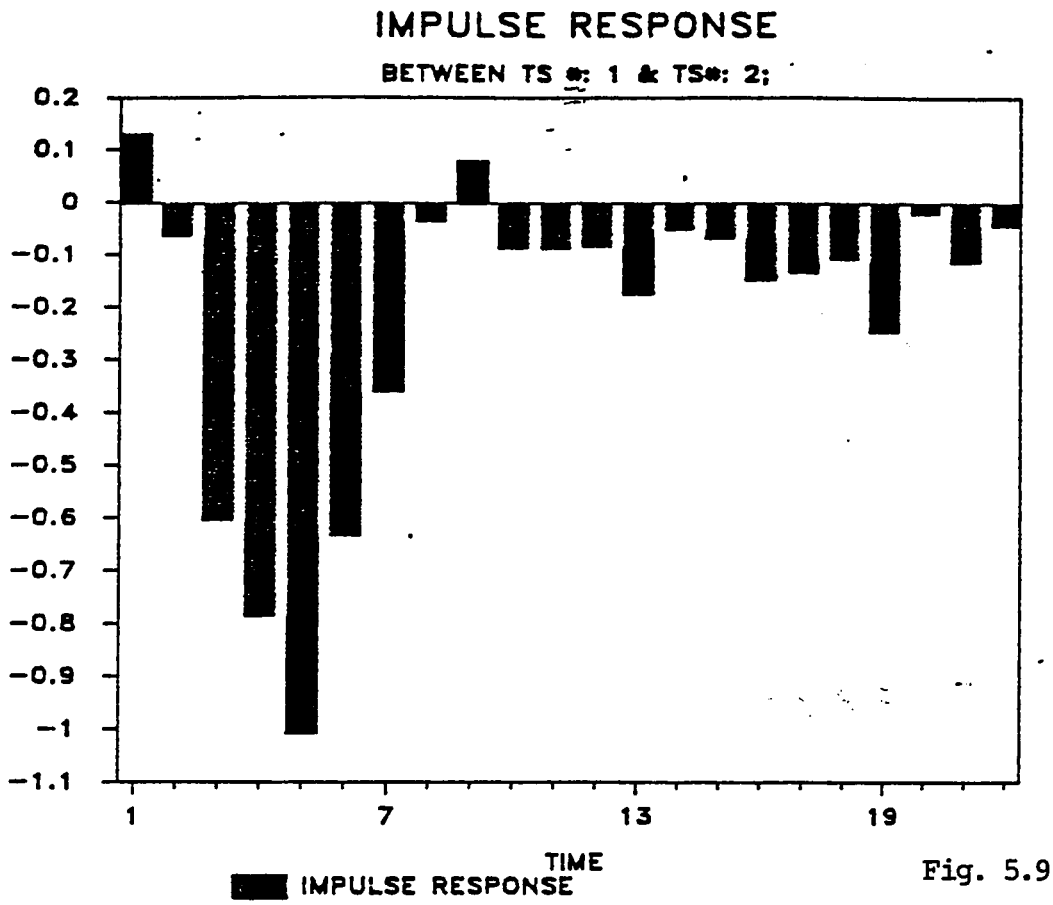


Fig. 5.9

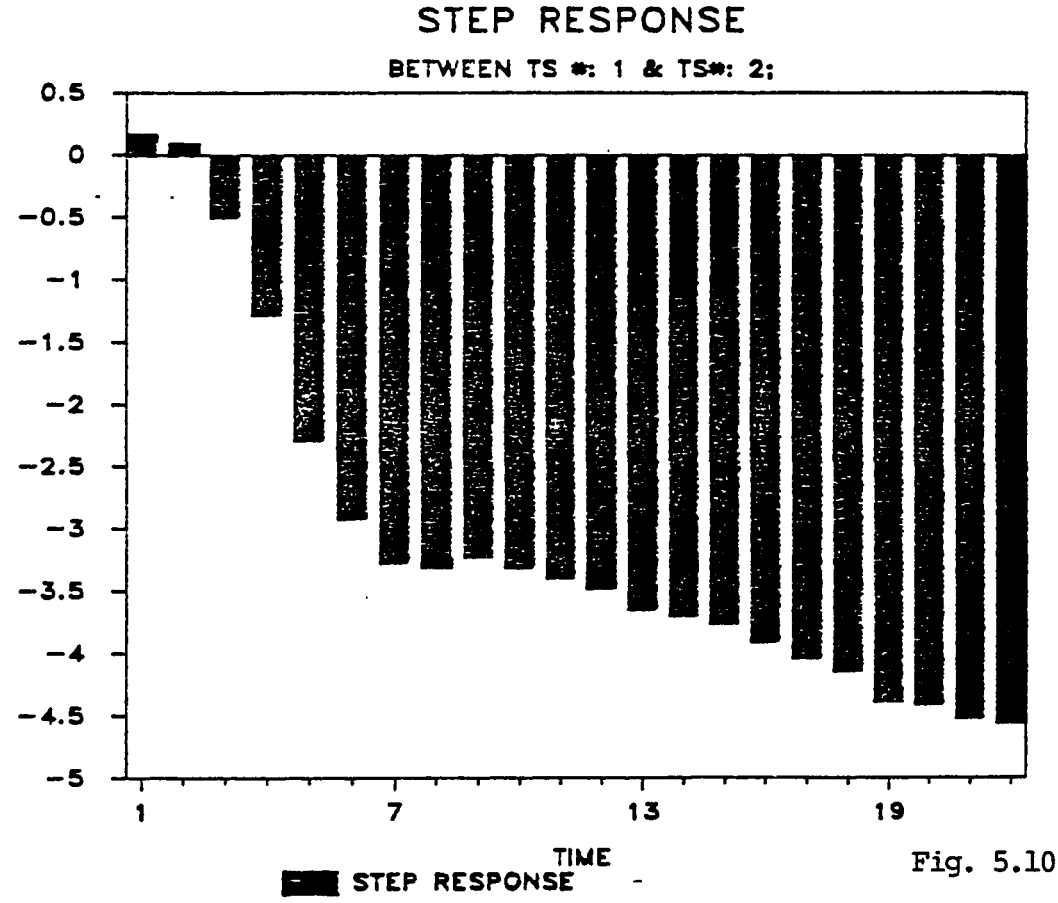


Fig. 5.10

(Fig-5.11) and subsequently it's estimated autocorrelation and partial autocorrelation are calculated and are plotted (Fig-5.12) to define the order of noise series ARMA model and to check for nonstationarity. The series is assumed to be stationary and since only first two partial autocorrelation functions are non-zero a second order autoregressive ARMA(2,0) is chosen. The Preliminary estimates routine gives the following noise model

$$(1 - 1.41\,B + 0.63\,B^2)\,N_t = a_t$$

The parameter vector θ is now assembled packed in the form.

$$\theta = (\omega_o, \omega_1, \omega_2, \delta_1, \delta_2, \phi_1, \phi_2)$$

$$= (-0.55, 0.33, 0.50, 0.56, 0.04, 1.4, 0.63)$$

and the nonlinear estimation routine is called with the parameters:

Step length = 0.1 ;

Increment factor for numerical derivative = 1×10^{-4};

Convergence criterion = 1×10^{-2}.

Using the micro computer the algorithm took 6 iteration to satisfy the convergence criterion and the elapsed time was

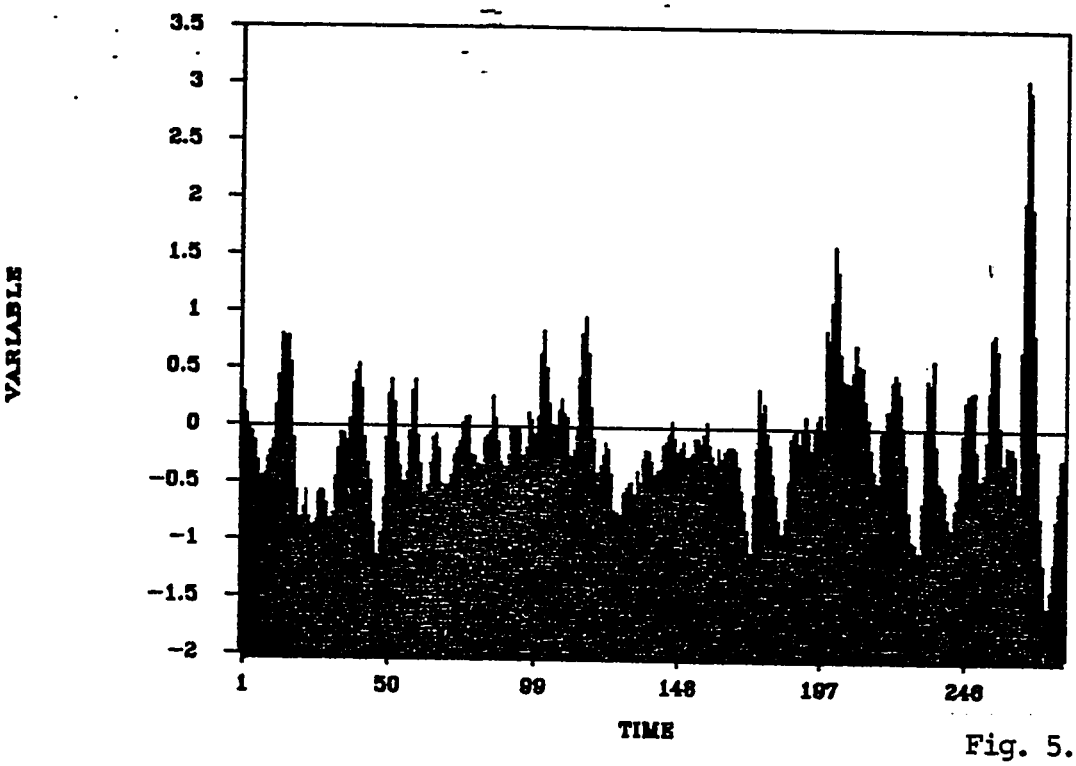


Fig. 5.11

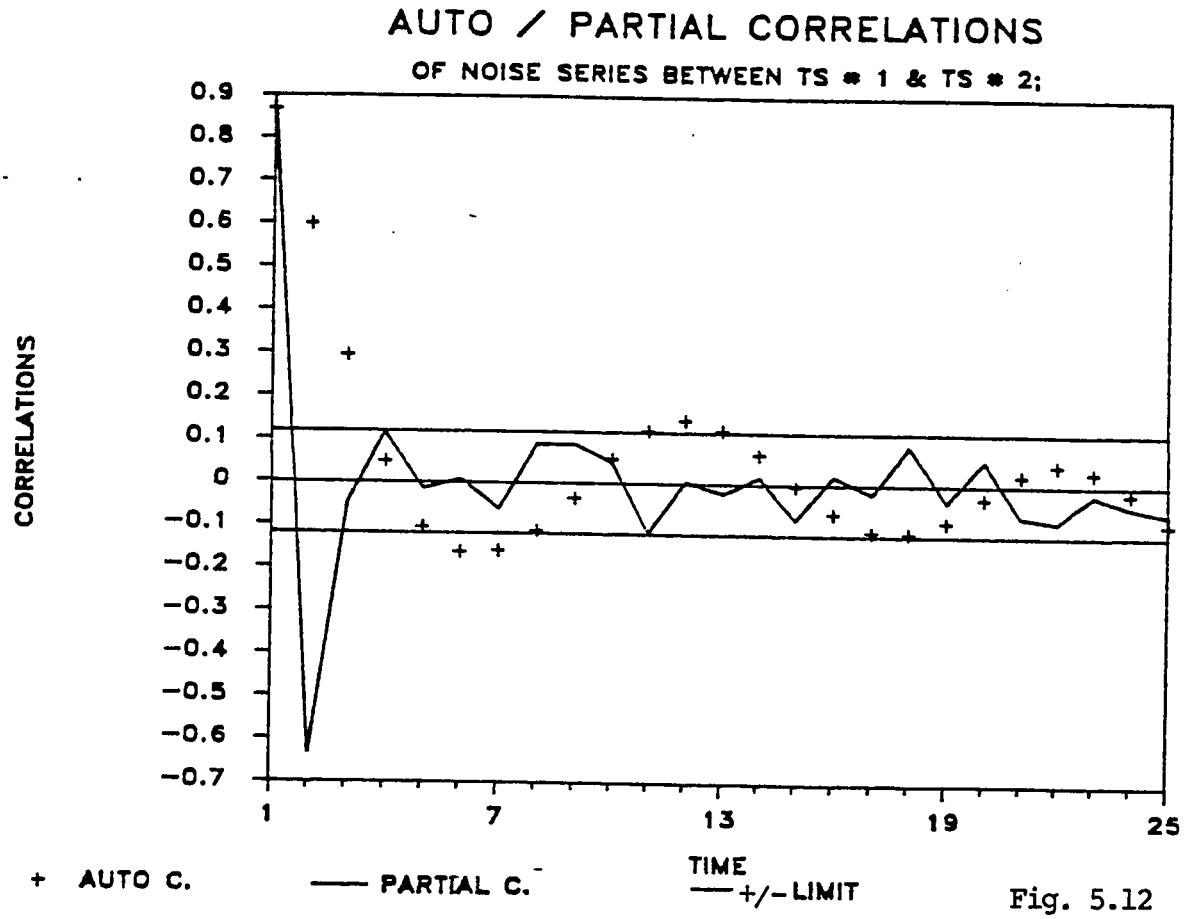


Fig. 5.12

noted as 95 seconds. The resulting final transfer function is

$$Y_t = - \frac{(0.53 + 0.39\ B + 0.51\ B^2)}{(1 - 0.54\,B + 0.001\,B)}\ X_{t-3}$$

with the fitted noise model given by

$$(1 - 1.5\,B + 0.64\ B^2)\ N_t = a_t \qquad (5.1.3)$$

Since the value of δ_2 parameter is negligible (0.001), the transfer function can be more conveniently represented by a (r,s,b) = (1,2,3) model to give the form:

$$Y_t = \frac{-(0.53 + 0.39\ B + 0.51\ B^2)}{(1 - 0.54\ B)}\ X_{t-3} \qquad (5.1.4)$$

The obtained model is now subjected to the diagnostic checks suggested by Box and Jenkins.

Firstly the residual autocorrelations are calculated and plotted (Fig-5.13) with their $\pm 2/\sqrt{N}$ limits. The correlation graph shows that all autocorrelation lie within the standard error limit band. To confirm the assumption the Q statistic test is applied (2.3.12) giving

$$Q = (296 - 2 - 3 - 2) \sum_{k=1}^{25} \hat{\gamma}_{aa}^{2}(k) = 26.7$$

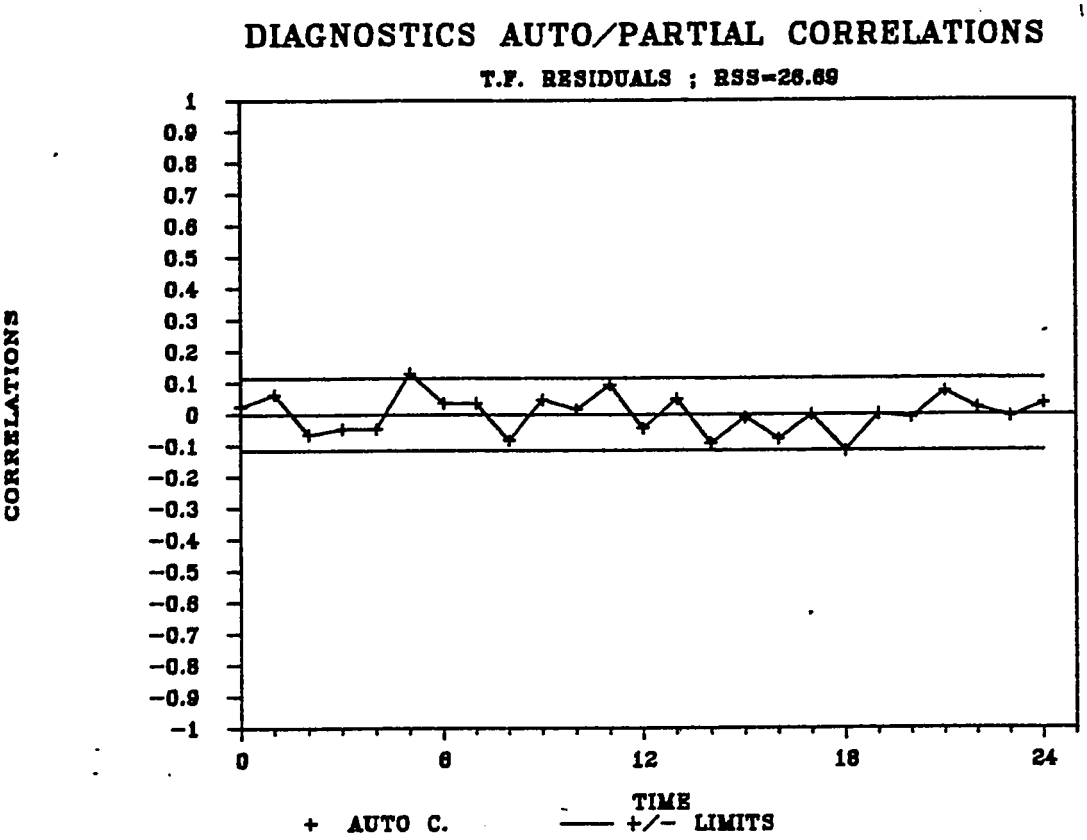

DIAGNOSTICS AUTO/PARTIAL CORRELATIONS
T.F. RESIDUALS ; RSS=26.69
CORRELATIONS
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
-0.8
-0.9
-1
0
6
12
18
24
TIME
+ AUTO C.
+/- LIMITS

Fig. 5.13

The x^2 distribution at a 5% significance level and with 25-2-0=23 degrees of freedom gives a value of 35.17. Therefore there are no grounds to reject the model.

Secondly the input and residual series cross correlation are computed and plotted (Fig-5.14) upto lag 25. Although the cross correlations are small they show a pronounced wave pattern, suggesting a relation between the input and the residuals. Box and Jenkins explained this by stating that for an adequate transfer function when the x_t's are autocorrelated, the cross correlations follow the same process as the input.

The last test involves the cross correlation between prewhitened input and the residuals (Fig-5.15). For the Q statistic

$$U = \max (s+b+p,\ p_x)$$

$$\max(2+3+2,3) = 7$$

which gives

$$Q = (296 - 7) \sum_{k=0}^{25} r_{\alpha\hat{a}}(k) = 19.8$$

The Chi-square distribution at 5% significance with k - r - s = 22 gives 33.92 suggesting adequacy of the model.

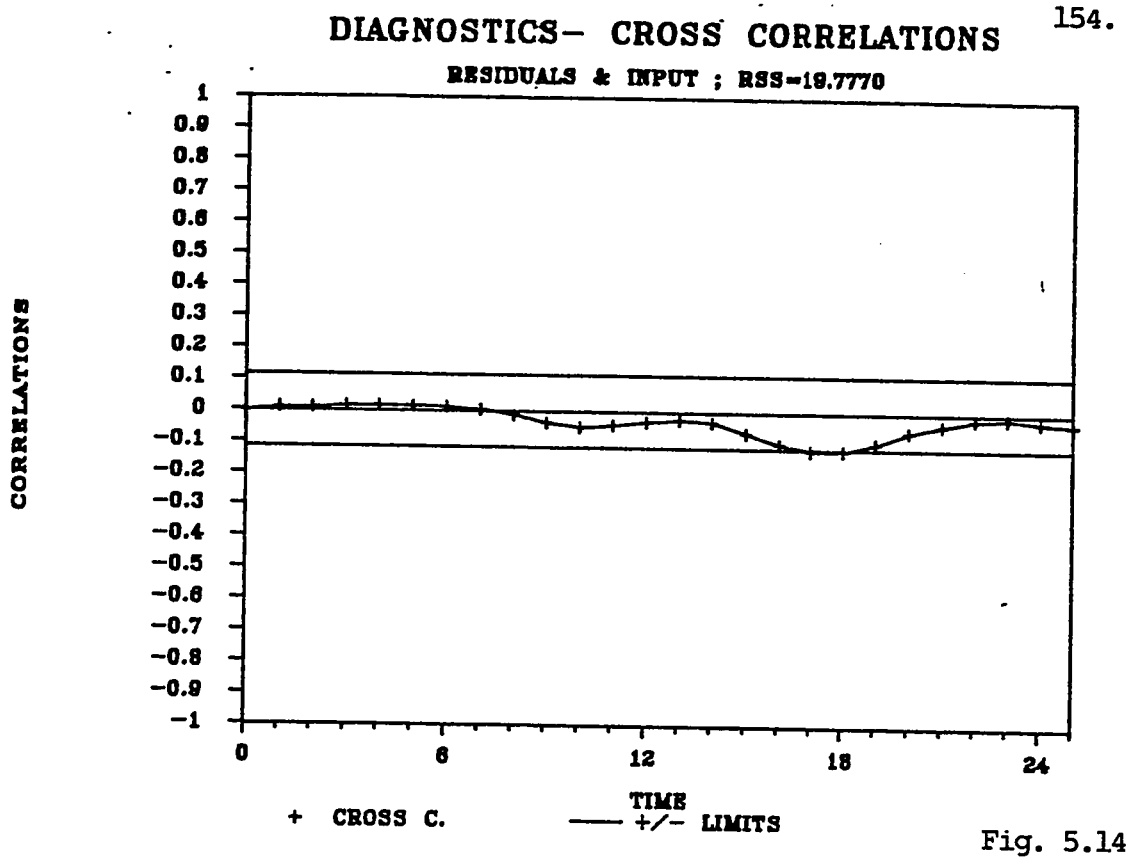


Fig. 5.14

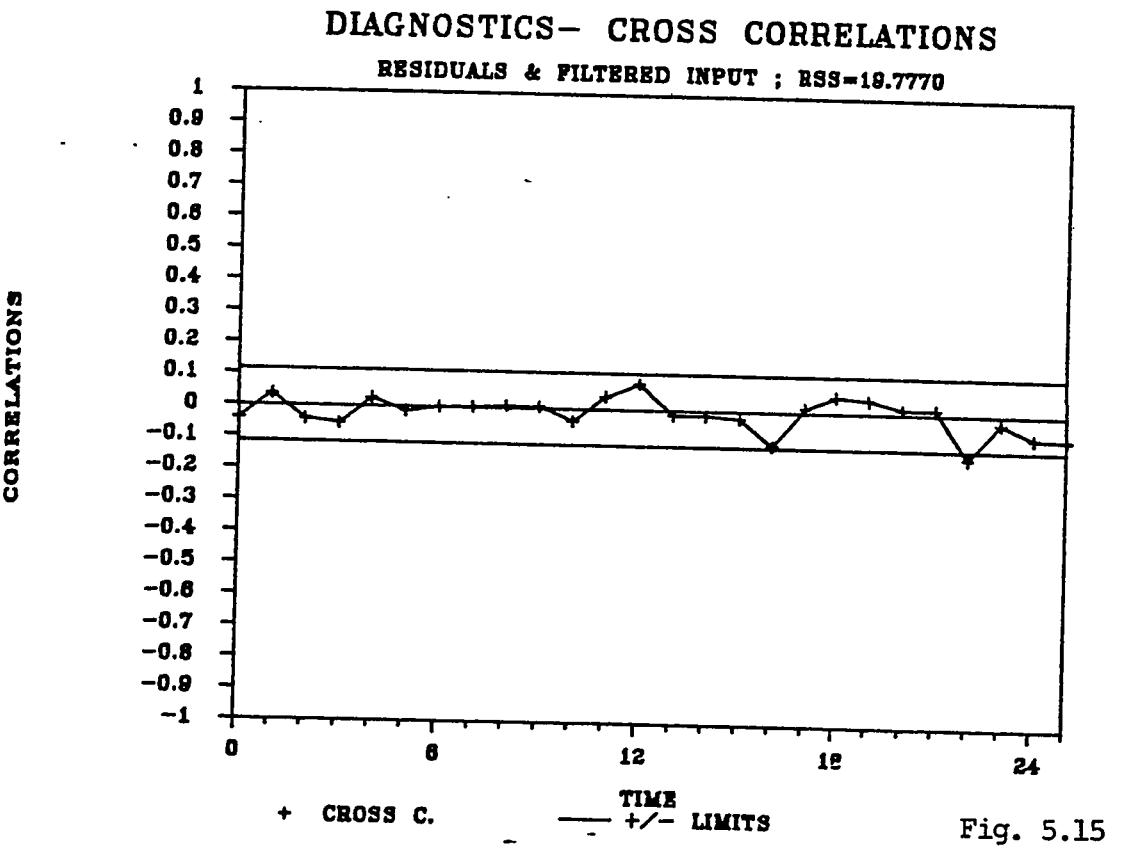


Fig. 5.15

The above transfer function is tested for stability, by requiring that the roots of the characteristic equation

$$\delta(B) = 0$$

For first-order model, the parameter satisfies

$$-1 < \delta_1 < 1$$

For the above transfer function model the condition for stability

$$-1 < 0.54 < 1$$

holds true.

The steady-state gain of the system is given by substituting the values of parameters in final transfer equation (5.1.4)

$$g = \frac{-(0.53 + 0.39 + 0.51)}{(1 - 0.54)} = -3.11$$

This means that if X_t is held indefinitely at a value +1, Y_t will eventually reach the value -3.11.

The Box-Jenkins modeling procedure is concluded by using the identified transfer function model for forecasting using the leading indicator method. Standing at time lag 277, seven step ahead forecasting is carried out (Fig-5.16). It is observed that the forecasts closely follow the real output and also all of the actual output data points lie within the 95% probability limits. It is also apparent that the

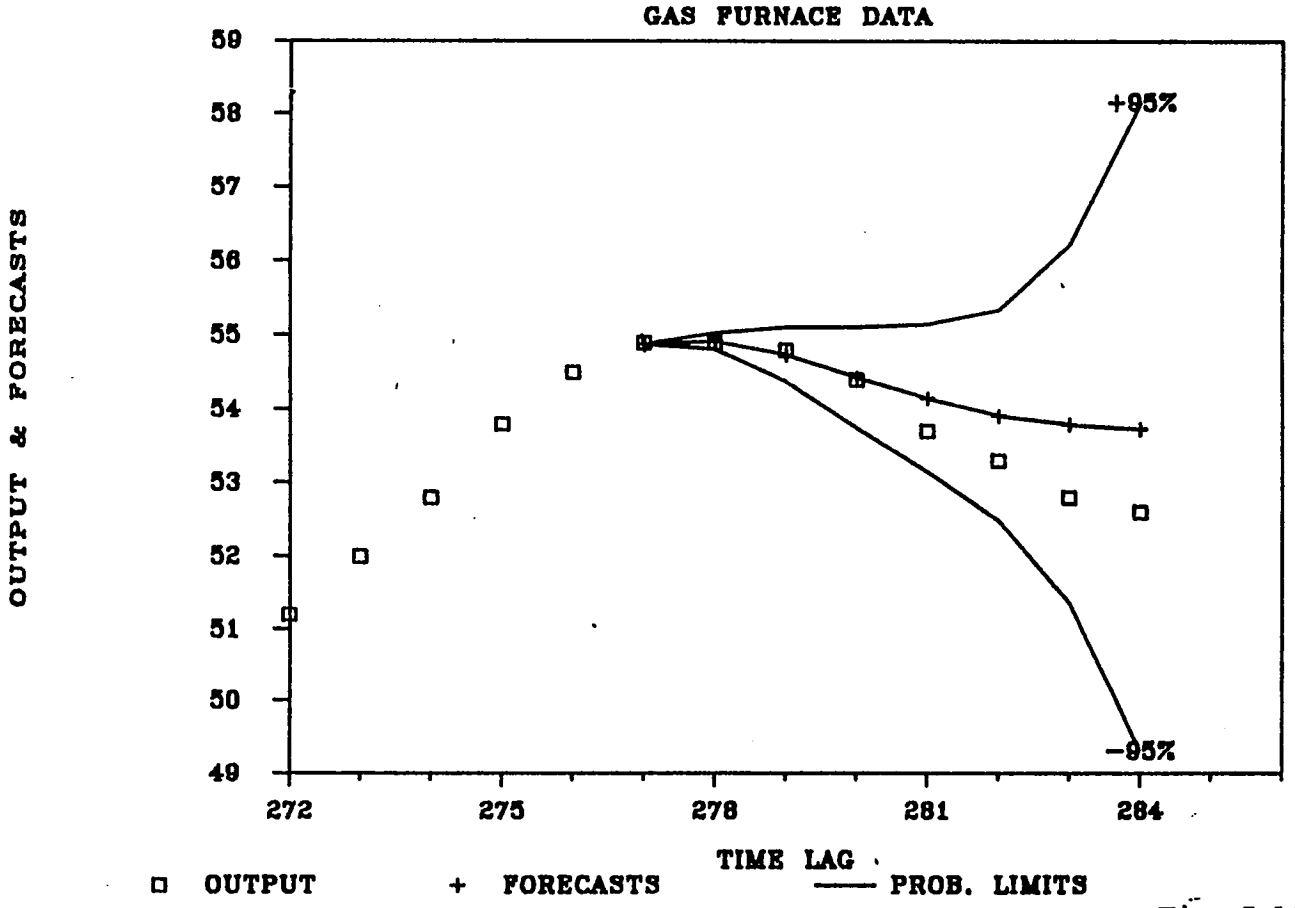

GAS FURNACE DATA
OUTPUT & FORECASTS
59
58
57
56
55
54
53
52
51
50
49
+95%
−95%
272
275
278
281
284
TIME LAG
□ OUTPUT
+ FORECASTS
—— PROB. LIMITS

Fig. 5.16

probability limits increase in width as the forecast lead time increases. The accuracy of the forecasts demonstrates accuracy and efficiency of the identified transfer function model.

5.1.2 Fitting Extended ARMA Models:

The gas furnace data is also used to find an Extended Auto-Regressive Moving Average (EARMA) modeling program on the microcomputer.

The first step in modeling EARMA models is to find the initial delay or dead time present in the system. Pandit and Wu suggest either using the initial values of the parameters obtained from the inverse function or to fit a high order pure autoregressive process for determining the system dead time. The input is coded as element 1 and output as element 2 of the EARMA vector and a autoregressive model of order 7 and 8 is fitted to both input and output.

Auto regressive parameters with respective confidence intervals are plotted (Fig.5.1.17 - 5.1.18). The plots (5.1.17 (a)+(b)) indicate that the dead time between X_{2t} and X_{1t} is three. On the other hand, there seems to be no relation of X_{2t} on X_{1t} as the confidence intervals of all the parameters in Fig. (5.18(a) + (b) include zero. This means that this is an open loop process with no feedback elements present.

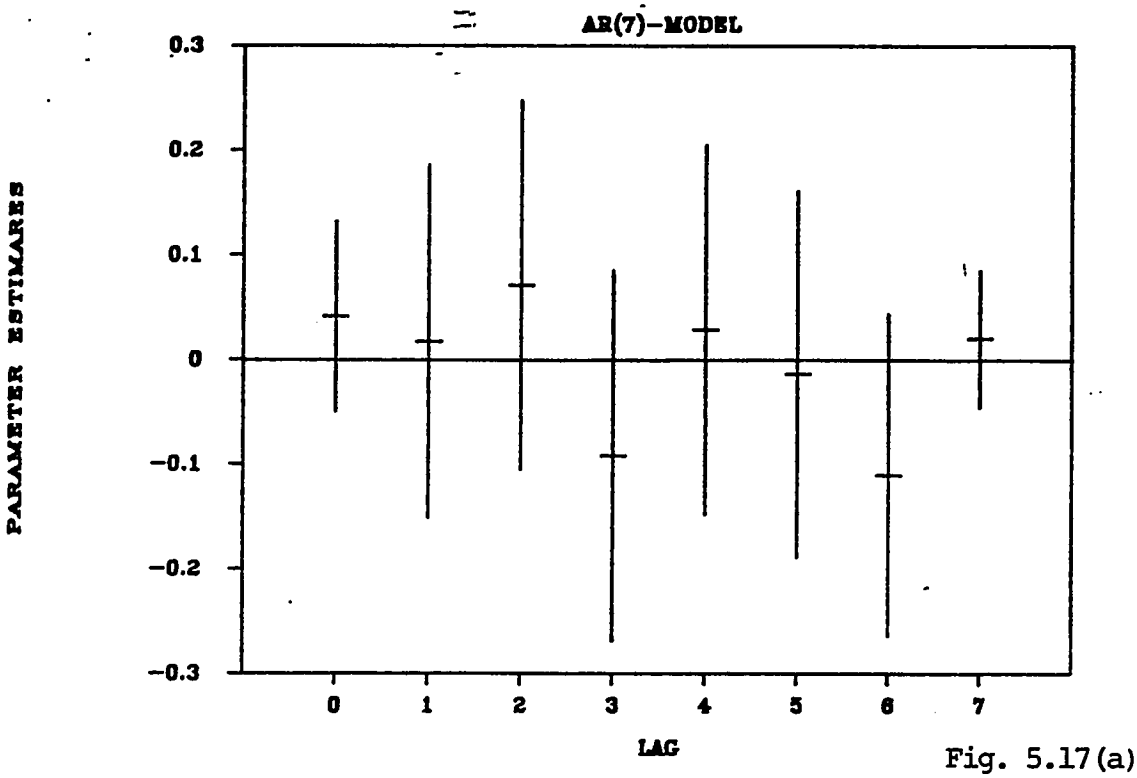


Fig. 5.17(a)

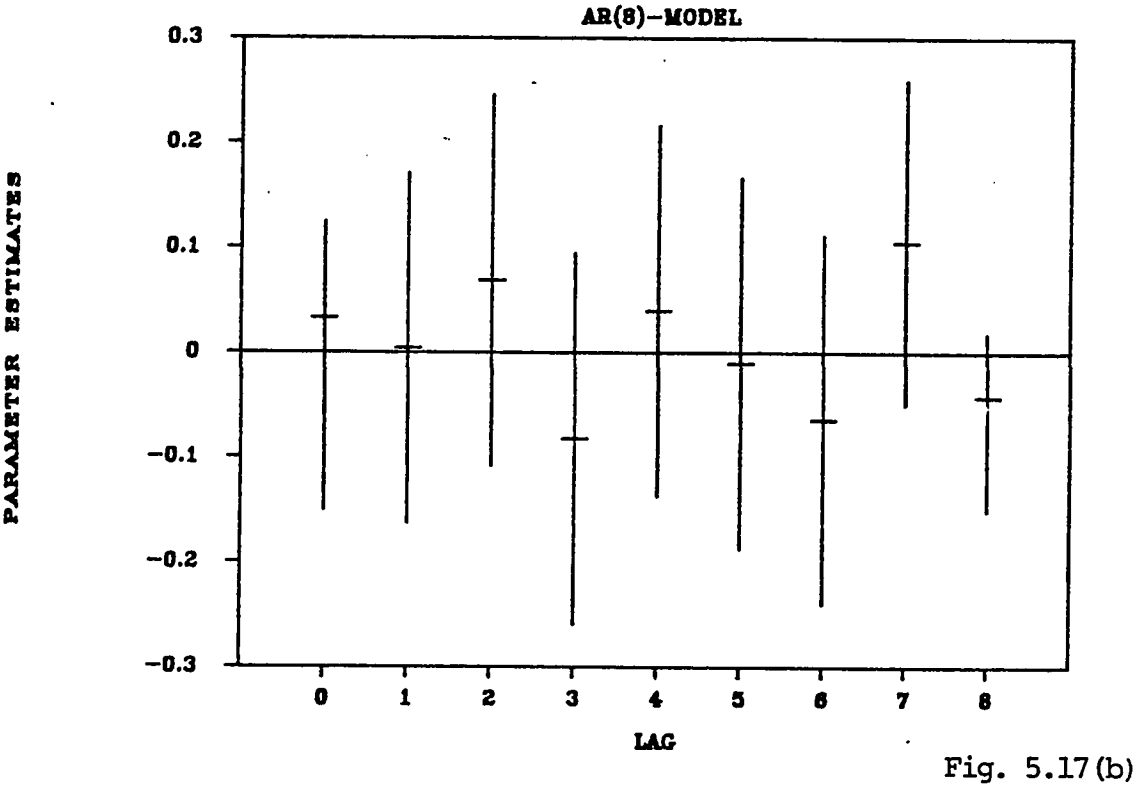


Fig. 5.17(b)

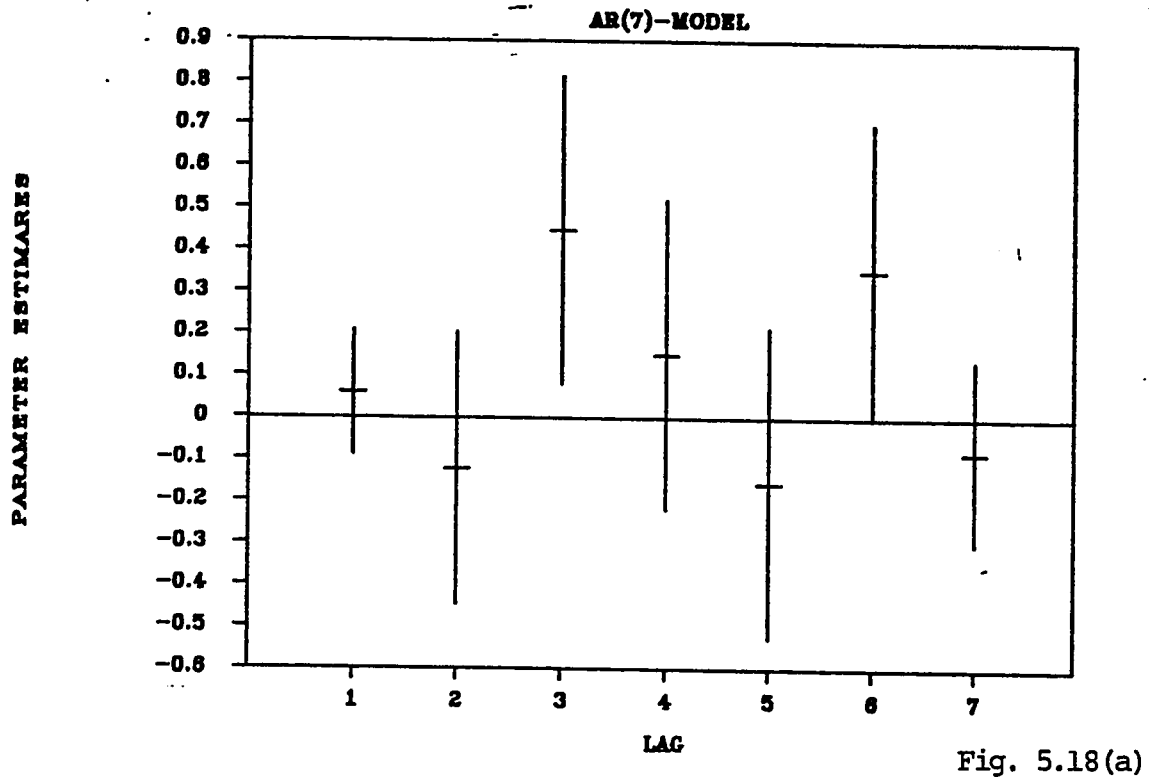


Fig. 5.18(a)

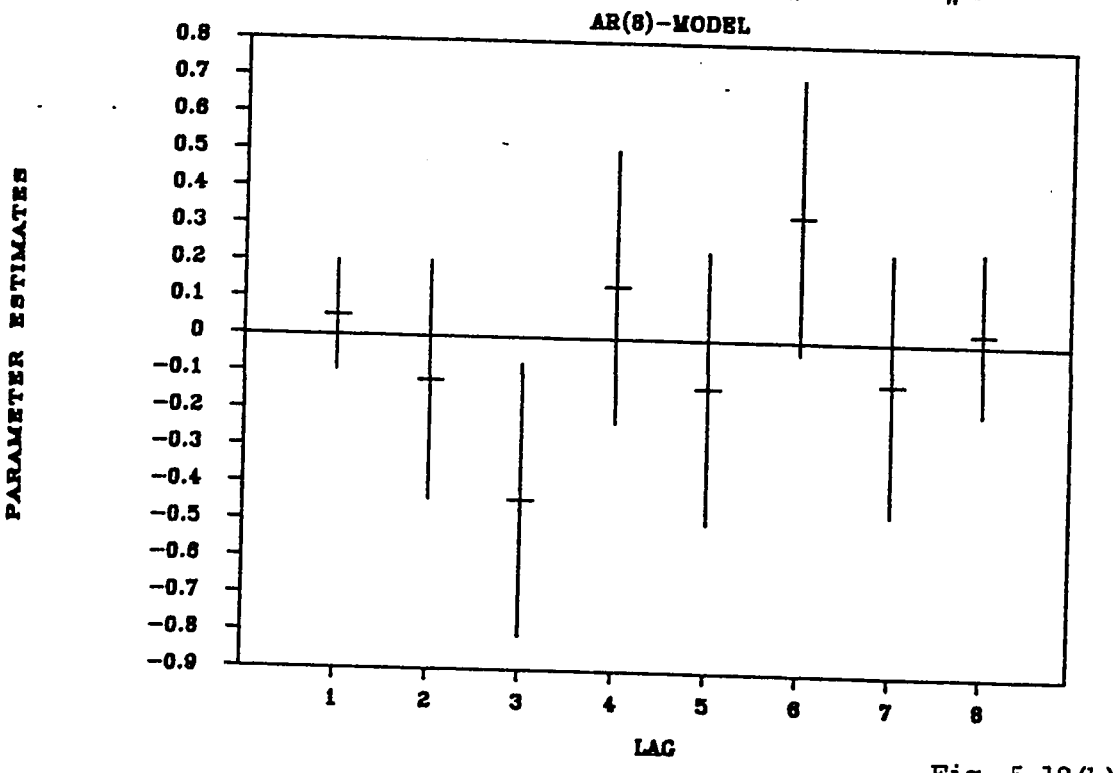


Fig. 5.18(b)

After the determination of the lags between the series, the EARMA modeling program is executed using the (n,n-1) approach. The results are summarized in Table 5.1 and Table 5.2. The program is run assuming that no a priori information other than system delay is available. Table 5.1 shows that the results of modeling the X_{1t} input series. For example, comparing AR(1,1,0) with ARMA(2,2,1).

$$F = \frac{22.75 - 10.30}{3} \div \frac{10.30}{296 - 5} = 116.71$$

Since the improvement is significant, increase n by one and fit an EARMA (3,3,2) model. The EARMA(3,3,2) model gives an F-value of 1.448 which is not significant at 5% level. Hence the ARMA(2,2,1) is an adequate model. However observing the EARMA(3,3,2) model closely reveals that as expected estimates of all moving average parameters θ_{221} and θ_{222} include zero in them, since this is an open loop process ϕ_{121}, ϕ_{122} and ϕ_{123} are close to zero. An EARMA(3,0,0) model is chosen as it contains no moving average parts and final estimates are obtained using a much simplified ARMA(3,0) model. The F-test used with the previous model shows that the model is adequate.

In Table 5.2, the X_{2t} output series is successively approximated by an EARMA(n,n-1) for n=1,2...4. Insignificance of the F-test is obtained in the EARMA(4,4,3) case, so an

TABLE 5.1

GAS FURNACE DATA: SERIES #1 -- INPUT GAS RATE.

PARAMETERS	MODEL ORDER			
	(1,0)	(2,1)	(3,2)	(3,0) *
PHI(1,2,0)	-.259 +/-.549	.037 +/-.086	.0543 +/-.090	.016 +/-.015
PHI(1,1,1)	1.124 +/-.043	1.730 +/-.121	2.129 +/-.674	1.936 +/-.113
PHI(1,1,2)		-.891 +/-.154	-1.684 +/-1.18	-1.260 +/-.217
PHI(1,1,3)			.479 +/-.618	.233 +/-.147
PHI(1,2,1)	-.202 +/-.050	-.001 +/-.127	.060 +/-.189	
PHI(1,2,2)		.121 +/-.154	-.001 +/-.168	
PHI(1,2,3)			-.016 +/-.065	
THT(1,1,1)		-.188 +/-.154	.214 +/-.673	
THT(1,1,2)			-.139 +/-.205	
RESIDUAL SUM OF SQUARES	22.75	10.30	10.15	10.28
F-TEST		116.71	1.45	1.85

* -THE ADEQUATE MODEL.

EARMA(3,3,2) is judged to be adequate. The confidence limits of the EARMA(3,3,2) parameters are wide and especially those of ϕ_{214} and ϕ_{215} include zero; therefore, there is a possibility that an EARMA(3,1,2) may also be adequate. Comparing the EARMA(3,1,2) model with the EARMA(3,3,2)model gives F = 2.98 where as $F_{0.05}(2,\infty)$ = 3.00. Hence the EARMA(3,1,2) model with the estimated parameters in the last column of Table 5.5 may also be considered adequate, although the F-test is clearly a border line case.

The step 4 of the Extended ARMA modeling algorithm is satisfied by calculating and plotting(Fig-5.19) the residual correlations. It is observed that the residual autocorrelation of the input (Fig-5.19 a), the residual autocorrelations of the output series (Fig-5.19 c) and the cross correlations between input and output residuals (Fig-5.19 b) all lie within the standard error limits. Hence the chosen models are accepted as the final models. The models can be expressed in difference equation form as:

$$X_t = -0.016\, X_{2t} + 1.94\, X_{1t-1} - 1.26\, X_{1t-1} + 0.233 X_{1t-3} + a_t \qquad (5.1.5)$$

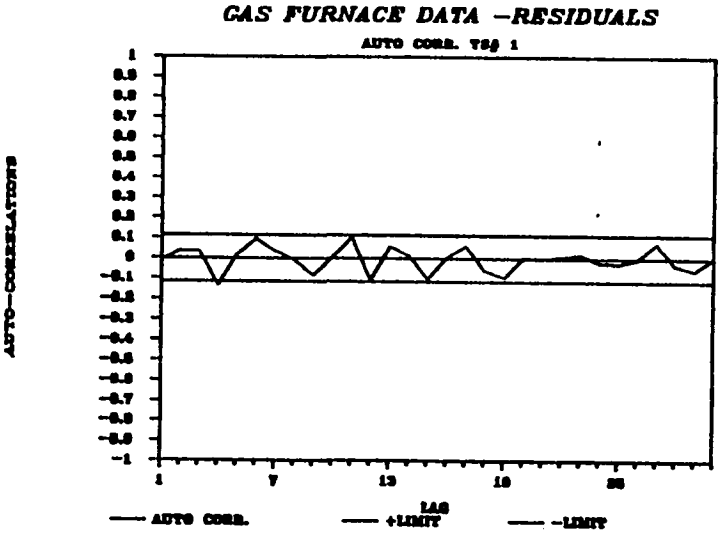
$$X_{2t} = -0.46\, X_{1t-1} + 1.65\, X_{2t-1} - 0.11\, X_{2t-2} + 0.304\, X_{2t-3} + a_{2t} - 1.510\, a_{2t-1} + 0.37\, a_{2t-2} \qquad (5.1.6)$$

TABLE 5.2

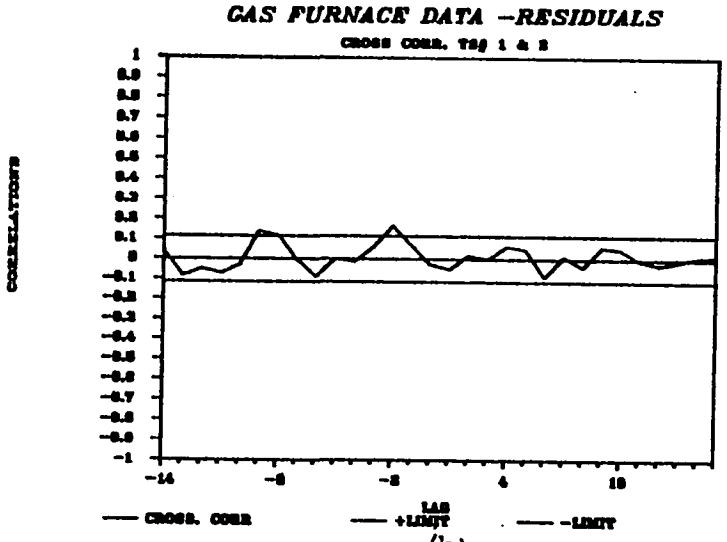
GAS FURNACE DATA: SERIES #2 -- % CO_2 IN OUTLET GAS

PARAMETERS	MODEL ORDER				
	(1,0)	(2,1)	(3,2)	(4,3)	(3,2) *
PHI(2,1,3)	.859 +/-0.06	-.672 +/-.131	-.468 +/-.136	-.544 +/-.144	-.459 +/-.081
PHI(2,1,4)		.218 +/-.180	.058 +/-.349	.394 +/-.426	
PHI(2,1,5)			.137 +/-.217	-.411 +/-.416	
PHI(2,1,6)				-.267 +/-.281	
PHI(2,2,1)	.765 +/-.020	1.379 +/-.111	1.827 +/-.269	1.96 +/-.662	1.648 +/-.183
PHI(2,2,2)		-.522 +/-.085	-1.24 +/-.383	-1.74 +/-1.20	-.109 +/-.283
PHI(2,2,3)			-.332 +/-.155	.895 +/-.876	.304 +/-.128
PHI(2,2,4)				-.204 +/-.262	
THT(2,2,1)		-.174 +/-.149	.290 +/-.281	.426 +/-.671	.151 +/-.201
THT(2,2,2)			-.522 +/-.085	-.543 +/-.284	-.366 +/-.118
THT(2,2,3)				.024 +/-.247	
RESIDUAL SUM OF SQUARES	44.24	18.30	16.43	16.0	16.77
F-TEST		137.6	10.89	2.522	2.99

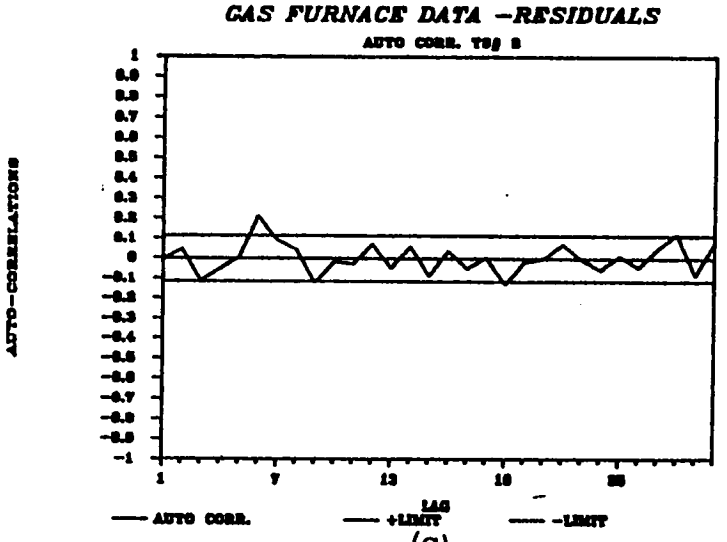
* -THE ADEQUATE MODEL.



(a)



(b)



(c)

Fig. 5.19 Auto/Cross Correlation of Residuals.

The modeling of gas furnace data was carried out using the Pandit and Wu EARMA(n,n-1) approach. Modeling was not attempted using the EARMA(2n,2n-1) method as it was found that execution time of the program increased in a roughly exponential manner as the EARMA model order is increased. After initial analysis ARMA(3,0) was chosen for the input X_1 and EARMA(1,3,2) for output X_2, the final model forms were entered and the estimates were obtained form the program in an execution time of about 6 minutes.

5.2 Analysis Using Multivariable Data

Frequently, in process control the relationships are multivariable i.e. an output variable is affected by more than one manipulated variable. Implying that effect of a single leading indicator on an output may not be significant. In modeling multivariable systems with Box and Jenkins technique one can expect transfer functions with a low order and size of right hand parameters and a high order significant noise model.

Multivariable data of a distillation column from one of the local process plants were obtained for analysis. The data consisted of three series of 288 observations each. Little information was available about the dynamics or input-output relationship of this data. In this context these time series

are referred to as X_{1t}, X_{2t} and X_{3t}.

Preliminary interactive analysis of this data was done using Box and Jenkins. X_{2t} was modelled as an ARMA(2,0) (Fig-5.20) process,

$$x_{2t} = 1.44x_{2(t-1)} - 0.485x_{2(t-2)} + a_t$$

but the sum of square of the final model had the relatively high value 558.24 and the resulting chi square test of residual auto correlations was accepted after initial doubts (Fig-5.21).

Prewhitening the X_{2t} with eq.(5.21) and transforming X_{3t} as output generated the cross correlation plots of (Fig-5.22). As suspected the influence of input on the output model was small and the preliminary transfer function model obtained was

$$x_{3t} = 0.346\, x_{2t} \qquad (5.2.2)$$

The noise series, describing the affect of variables other than X_{2t} on output X_{3t}, was found to have non-stationarities and a high ARMA order (Fig-5.23).

In view of the high sum of square's of residual's and multivariable nature of data, extended ARMA models were called for. Assuming no apriori information was available ,

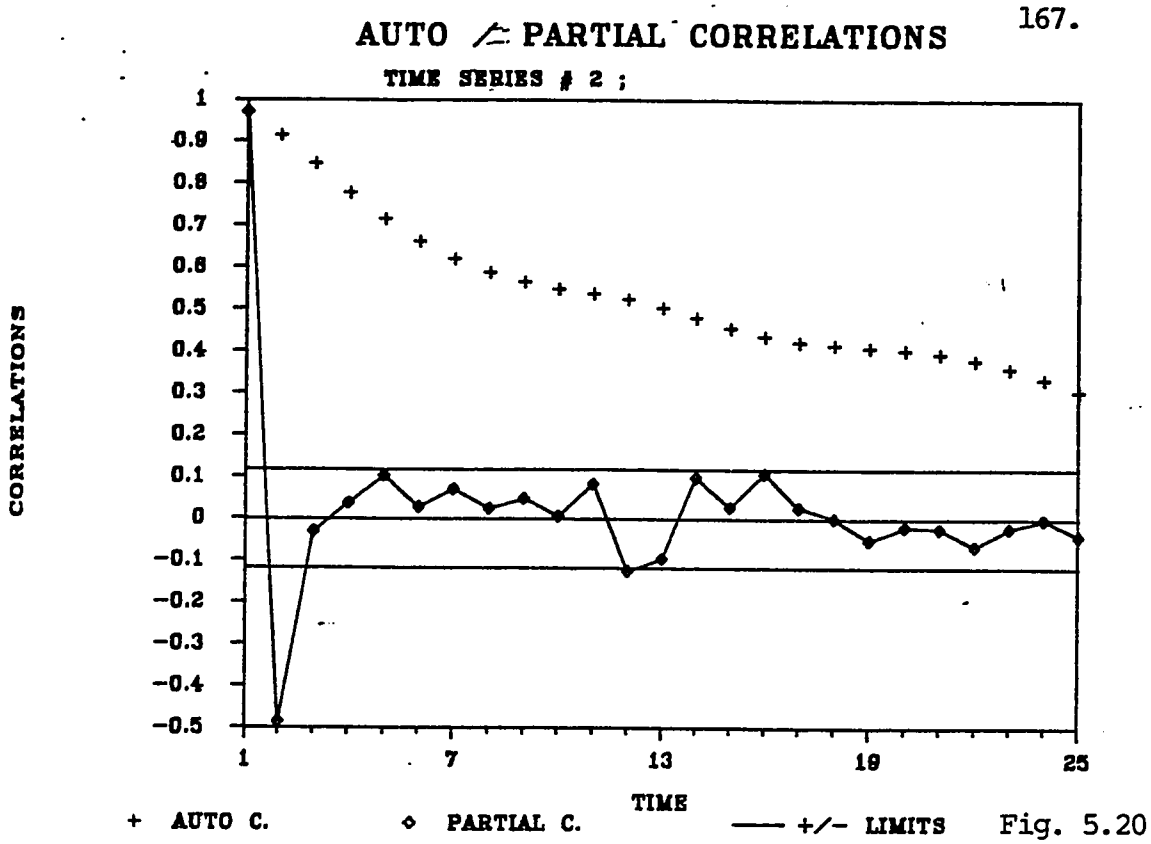


Fig. 5.20

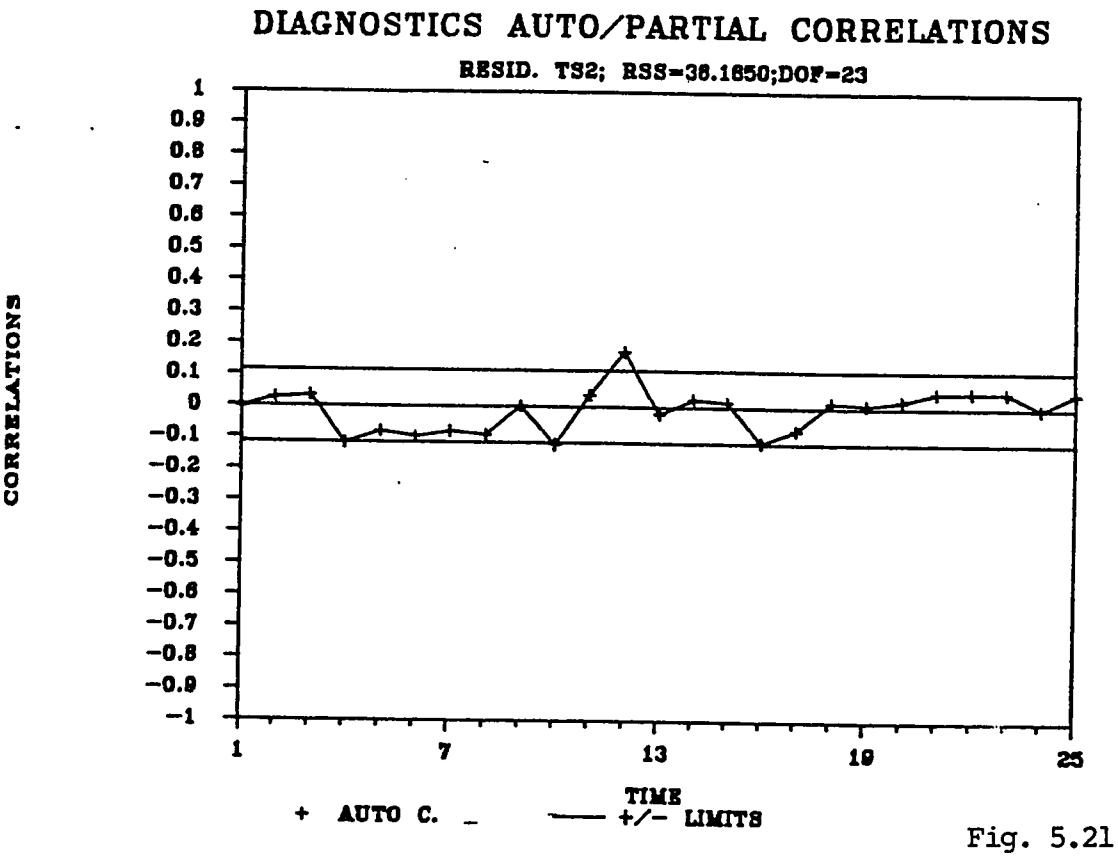


Fig. 5.21

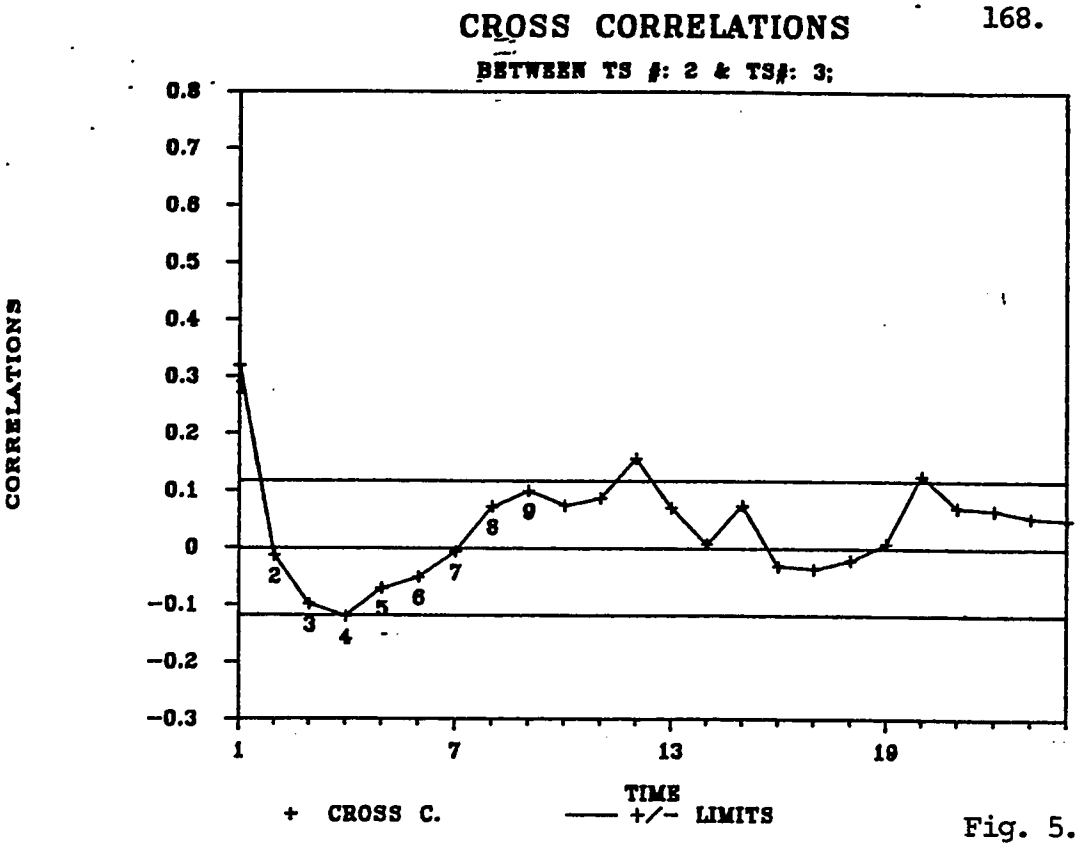


Fig. 5.22

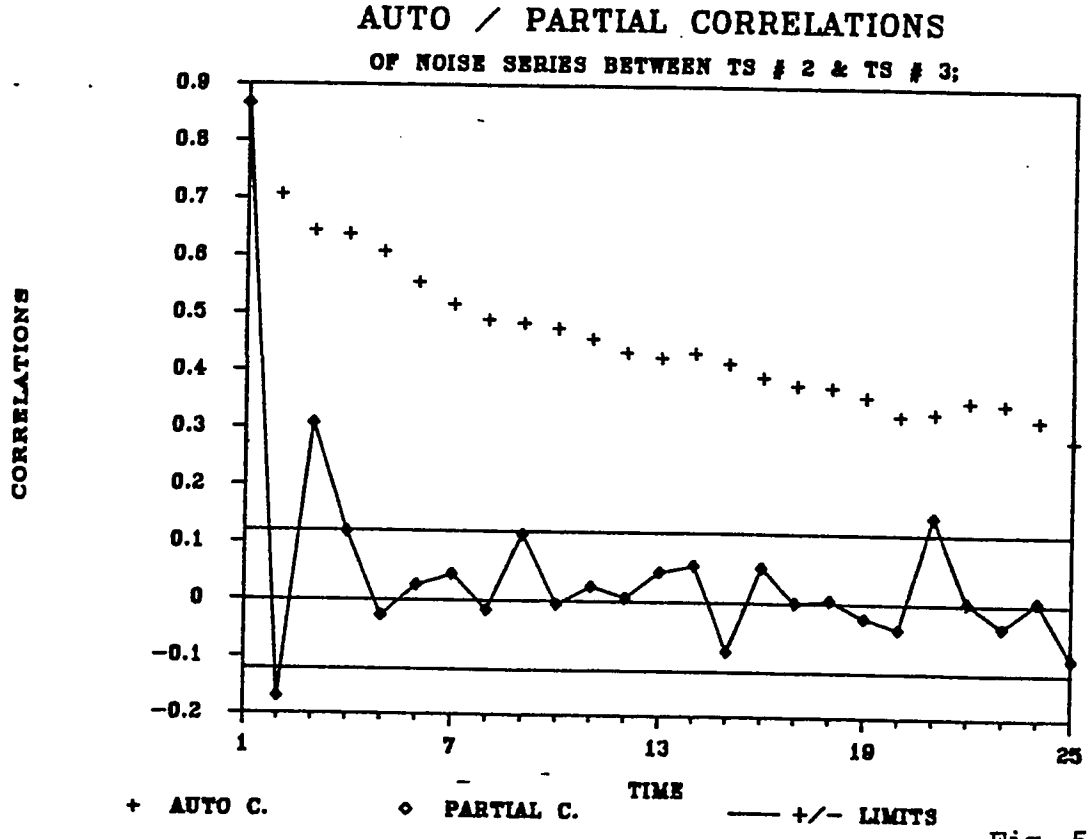


Fig. 5.23

the batch program of Pandit and Wu using (n,n-1) strategy was executed for the distillation data for n=1,2,3,4. The results are summarized in Table 5.3,5.4 and 5.5. Table 5.3 for X_{1t} indicates that adequacy of the model is attained for EARMA (3,3,3,2) after EARMA(4,4,4,2) models gave insignificant reduction in the sum of squares. Small values for $\phi_{12\ell}$ and $\phi 13\ell$ sequences indicates that X_{1t} is dependent upon X_{2t} and X_{3t} and it also has distinctive $\phi_{11\ell}$ parameters. The confidence interval of the estimates indicates that ϕ_{333} can be made zero. The final model for X_{1t} is EARMA(3,3,2,2) and its final estimates are shown in table. (5.3)

Results of modeling X_2 shown in table (5.4) indicate adequacy of the EARMA(2,2,2,1) model . However all parameters of the $\phi_{21\ell}$ and $\phi_{23\ell}$ sequence include zero in their respective confidence intervals. A final run was made using EARMA(0,2,0,1) model. Table 5.4 shows significant reduction in the residual sum of squares for X_{2t} as compared to that of the Box Jenkins approach.

Similarly X_{3t} was analysed. The identified, adequate EARMA(2,2,2,1) model can be reduced to EARMA(0,2,1,1) and checked using the F test.

The diagnostic checks for residuals for all series were performed and are summerized in fig (5.24). Adequacy of the identified model orders is established by studying these

TABLE 5.3

DISTILLATION COLUMN DATA: SERIES #1

PARAMETERS	MODEL ORDER				
	(1,0)	(2,1)	(3,2)	(4,3)	(3,2) *
PHI(1,2,0)	-.005 +/-.036	.049 +/-.032	.073 +/-.030	.074 +/-.031	.074 +/-.030
PHI(1,3,0)	-.109 +/-.032	.063 +/-.033	.046 +/-.031	-.049 +/-.031	.047 +/-.029
PHI(1,1,1)	.894 +/-.049	1.379 +/-.154	1.988 +/-.196	1.277 +/-1.08	1.989 +/-.193
PHI(1,1,2)		-.543 +/-.145	-1.477 +/-.317	-.068 +/-2.11	-1.477 +/-.308
PHI(1,1,3)			.444 +/-.151	-.581 +/-1.59	.444 +/-.146
PHI(1,1,4)				.298 +/-.502	
PHI(1,2,1)	.006 +/-.032	-.073 +/-.048	.108 +/-.064	.057 +/-.095	.108 +/-.064
PHI(1,2,2)		-.027 +/-.030	-.027 +/-.071	.105 +/-.125	.027 +/-.070
PHI(1,2,3)			-.067 +/-.034	-.051 +/-.061	-.067 +/-.033
PHI(1,2,4)				-.045 +/-.082	
PHI(1,3,1)	.092 +/-.042	-.086 +/-.046	.150 +/-.057	.123 +/-.069	.150 +/-.049
PHI(1,3,2)		.002 +/-.036	-.086 +/-.059	.009 +/-.177	-.086 +/-.058
PHI(1,3,3)			.000 +/-.032	-.046 +/-.118	
PHI(1,3,4)				-.009 +/-.036	
THT(1,1,1)		-.135 +/-.183	.580 +/-.231	.139 +/-.011	.581 +/-.228
THT(1,1,2)			-.619 +/-.169	.333 +/-.643	-. 61 +/-.165
THT(1,1,3)				.005 +/-.204	
RESIDUAL SUM OF SQUARES	466.24	278.07	229.88	229.14	229.88
F-TEST		47.12	14.41	0.31	-

* -THE ADEQUATE MODEL.

TABLE 5.4

DISTILLATION COLUMN DATA: SERIES # 2

PARAMETERS	MODEL ORDER			
	(1,0)	(2,1)	(3,2)	(2,1) *
PHI(2,3,0)	-.741 +/-.117	-.297 +/-.110	-.269 +/-.115	-.307 +/-.048
PHI(2,1,1)	-.317 +/-.157	.017 +/-.288	-.045 +/-.427	
PHI(2,1,2)		-.114 +/-.298	-.056 +/-.699	
PHI(2,1,3)			-.051 +/-.436	
PHI(2,2,1)	.879 +/-.027	1.633 +/-.136	1.193 +/-.666	1.625 +/-.077
PHI(2,2,2)		-.715 +/-.117	.008 +/-1.07	-.716 +/-.072
PHI(2,2,3)			-.321 +/-.475	
PHI(2,3,1)	.404 +/-.127	-.033 +/-.180	.240 +/-.242	
PHI(2,3,2)		-.053 +/-.131	-.172 +/-.196	
PHI(2,3,3)			.070 +/-.153	
THT(2,2,1)		-.483 +/-.203	.004 +/-.658	0.438 +/-.144
THT(2,2,2)			.295 +/-.333	
RESIDUAL SUM OF SQUARES	499.15	354.27	347.31	362.80
F-TEST		28.63	1.38	1.68

* -THE ADEQUATE MODEL.

TABLE 5.5

DISTILLATION COLUMN DATA: SERIES # 3

PARAMETERS	MODEL ORDER			
	(1,0)	(2,1)	(3,2)	(2,1) *
PHI(3,1,1)	-.014 +/-.155	-.297 +/-.387	-.263 +/-.458	
PHI(3,1,2)		-.159 +/-.380	.138 +/-.714	
PHI(3,1,3)			-.007 +/-.429	
PHI(3,2,1)	.010 +/-.027	-.222 +/-.111	-.205 +/-.125	-.218 +/-.107
PHI(3,2,2)		-.261 +/-.101	.182 +/-.256	.246 +/-.098
PHI(3,2,3)			-.007 +/-.429	
PHI(3,3,1)	.832 +/-.081	.846 +/-.231	1.072 +/-.813	.765 +/-.113
PHI(3,3,2)		-.101 +/-.225	-.332 +/-.748	
PHI(3,3,3)			.107 +/-.247	
THT(3,3,1)		-.333 +/-.231	-.076 +/-.812	-.435 +/-.124
THT(3,3,2)			.095 +/-.362	
RESIDUAL SUM OF SQUARES	489.73	378.17	369.67	383.77
F-TEST		20.72	1.59	1.38

* -THE ADEQUATE MODEL.

residual correlation graphs. The adequate models for the series X_{1t}, X_{2t} and X_{3t} can be described in state variable representation.

$$\begin{bmatrix} X_{1t} \\ X_{2t} \\ X_{3t} \end{bmatrix} = \begin{bmatrix} 1.99 & 0.108 & 0.15 \\ 0 & 1.63 & 0 \\ 0 & -0.218 & 0.765 \end{bmatrix} \begin{bmatrix} X_{it-1} \\ X_{2t-1} \\ X_{3t-1} \end{bmatrix} + \begin{bmatrix} -1.48 & 0.027 & -0.086 \\ 0 & -0.716 & 0 \\ 0 & 0.246 & 0 \end{bmatrix} \begin{bmatrix} X_{1t-2} \\ X_{2t-2} \\ X_{3t-2} \end{bmatrix}$$

$$+ \begin{bmatrix} 0.444 & -0.067 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} X_{1t-3} \\ X_{2t-3} \\ X_{3t-3} \end{bmatrix} + \begin{bmatrix} a_{1t} \\ a_{2t} \\ a_{3t} \end{bmatrix} \qquad (5.2.3)$$

$$- \begin{bmatrix} 0.581 & 0 & 0 \\ 0 & 0.438 & 0 \\ 0 & 0 & -0.435 \end{bmatrix} \begin{bmatrix} a_{1t-1} \\ a_{2t-1} \\ a_{3t-1} \end{bmatrix} - \begin{bmatrix} -0.61 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_{1t-2} \\ a_{2t-2} \\ a_{3t-2} \end{bmatrix}$$

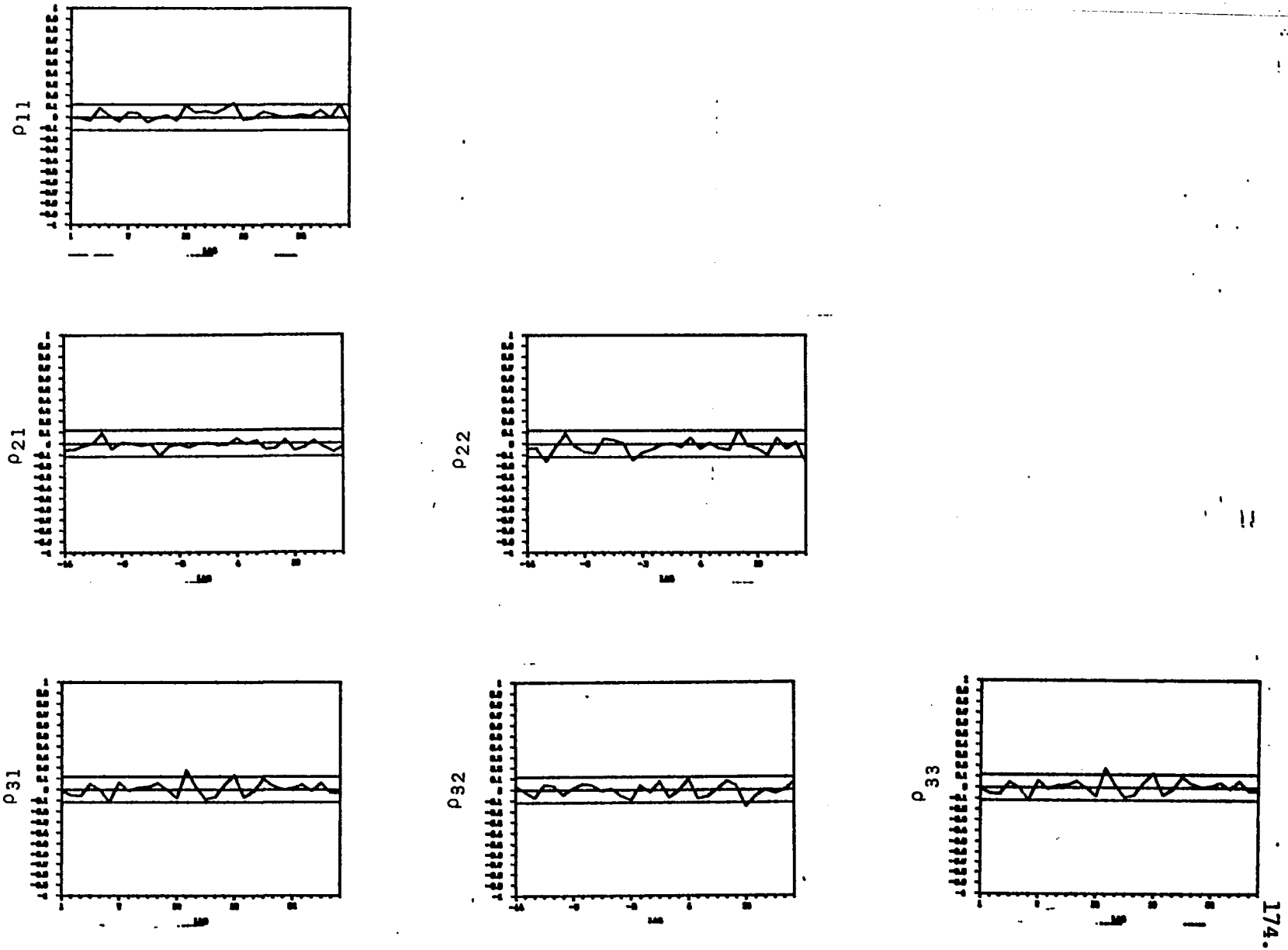


Fig. 5.24 Sample autocorrelation functions (diagonal plots) and sample cross-correlation functions (off diagonal plots) of Distillation Column Residuals.

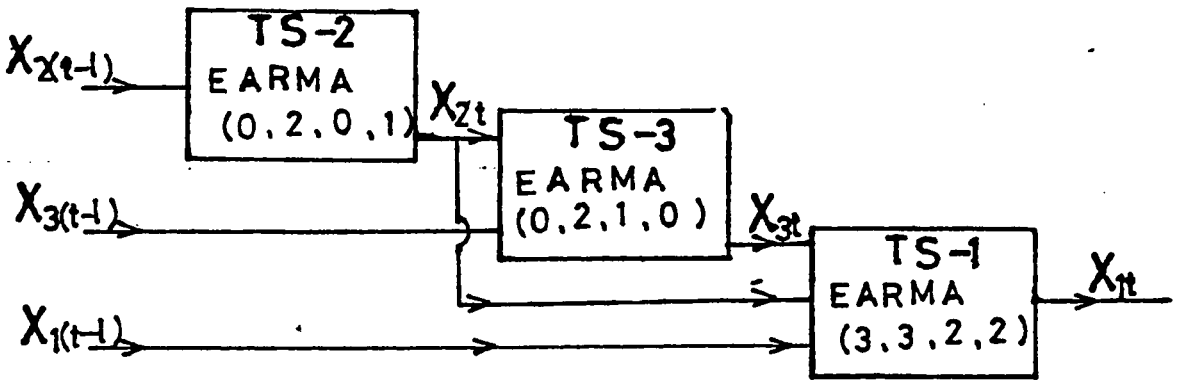


Fig-5.25

Interactions between X_{1t}, X_{2t} and X_{3t}

From the models it is observed that X_{2t} is not dependent on other series, while it affects both the X_{1t} and X_{3t} time series. Series X_{3t} although depending on X_{2t}, also influences X_{1t}. The X_{1t} series is perturbed by both X_{2t} and X_{3t}. The results show existence of an open-loop system. The system interactions are shown in (Fig-5.25).

It should be noted that modeling all time series is necessary to check the justification of using the EARMA models. A major advantage of using EARMA for this multivariable data was that a comprehensive knowledge of the system including open and closed loop structure and interaction between the inputs was obtained.

5.3 Microcomputer Experiences

In a developmental project such as this, it is considered essential to narrate some prominent achievements and some pitfalls encountered while working extensively with a microcomputer. Although this experience depends on the specific microcomputer and peripherals, the present comments should serve as useful guidelines for future users.

The main programming instrument used in this work is the Microsoft FORTRAN compiler ver.3.20 which is a subset of full FORTRAN 77. Although the FORTRAN compiler is one of the most powerful FORTRAN compilers of microcomputers it has limitations and some errors are apparent.

The biggest handicap encountered while working with MS-FORTRAN was the excessive amount of time needed for compilation as opposed to the mainframe FORTRAN. The compilation process of MS-FORTRAN consists of 2-3 passes, the function of which is to create the object code of the FORTRAN program with the aid of intermediate files. Next a linker program combines the object modules of several FORTRAN subroutines to create the execution file. Although the use of a hard disk, decreased the compilation time greatly over that based on a floppy disk drive, the time to complete pass1, pass2 and the linker was still considerable. The RAM disk facility created the object module in less time but because of limited size, large programs cannot be compiled and linking is not possible if many object modules are present

for one main program. Table 5.6 illustrates compilation times for a typical FORTRAN program using three different storage media (floppy disk, hard disk and RAM disk) for microcomputer compared to the compilation time on an IBM main frame. The program consisted of 38 lines with a subroutine of 304 lines.

Table 5.6 Fortran Compilation Time Using Different Configurations

	RAM Disk	Hard Disk	Floppy disk	Main Frame
Load for 1	3	5	11	
Run pass 1	30	36	53	
Load pass 2	3	6	12	
Run pass 2	80	105	155	
Load link	2	3	6	0.1 sec
Run linker	48	51	111	
Total	166 sec	206 sec	327sec	

Although the above times are approximate because of the nonavailability of a good timing device, it clearly shows the

amount of time and effort expended in compiling FORTRAN programs on the microcomputer as compared to the main frame computer.

The above problem is aggravated by the fact that MS-FORTRAN provides inadequate diagnostic messages. Furthermore some compiler bugs are apparent. The longest compiler message is eight words long and moreover the line number of an error is not displayed unless the $DEBUG metacommand is used, which generates a larger object code. A major problem faced by MS-FORTRAN users is that use of a DIMENSION or common statement in the program will set aside space in memory for any named variables during execution and after the end of execution this space will not be reset. Another run of the program will start with the old values of the variables unless the values are initialized in the beginning of the program or the computer is reset [36]. The bugs and limitations found in the compiler are few and not serious. After a call to some subroutines, the compiler did not accept the 'write using' FORMAT statement. In some programs the computer stalled with no error message. Moreover it was found that stock overflows occur fairly quickly while carrying out polynomial division, and during write operation to a number of files or devices.

On average six to ten times improvement in execution speed was obtained using the 8087 coprocessor but in Read/Write operations there was little improvement. Table 5.7

compares execution speeds for the leading indicator forecasting program with and without the 8087 processor, Main frame timings are also shown.

Table 5.7 Fortran Execution Time Using Different Configurations

	PC with 8087	PC without 8087	Main frame
Execution time	20 sec	142 sec	0.1 sec

The IBM-PC provides a direct link for transfering data between itself and the main frame using the IRMA card facility. Thus main frame FORTRAN programs can be made to run on the IBM-PC with minor modifications. The main difference was found in free-formatting statements such as PRINT, character manipulation, data statement, structured FORTRAN and representation of hexadecimal numbers. It was found that a considerable deviation of results can occur in programs where the round-off error in a 16-bit machine is significant or where the programs are machine dependent, such as some optimization routines and random number generator programs.

Although subroutines were used in the work to make the programs compact, modular and concise, it was found that considerable adaptation was necessary in the main program to integrate it with data entry, display and other program files. These integration tasks were frequently time-consuming and involved two to three times the effort required for

performing the subroutine algorithm alone.

The Symphony package was found to work well for integrating the different elements in the package, but its large size is a problem. Symphony requires 320K alone to become resident with the operating system in the RAM. The available memory left is thus 512-320=192 K. This 192 K had to be used efficiently so that it could be split, to perform the application written in the Symphony command language, to graph the large data sets of time series and to execute the complex time series modeling programs. This limitation can be removed to some extent by using a memory addon board with a capability of 384K giving a total memory of 640 K.

Symphony offers six diverse applications each having its own special commands coupled with its complete command language. Programming Symphony requires detailed experimentation and exploration to understand their full capabilities. However, Symphony also has some errors. Symphony offers a DOS window which can perform most of the operating system commands, but executing the PRINT command in DOS can destroy Symphony's memory; running batch files from DOS window will cause the system to stall. In some complex calculation's Symphony was found to be slow and gave inaccurate results; moreover, it was found to take no advantage of the installed 8087 coprocessor.

The graphic capabilities of Symphony allow for simultaneous display of seven colors or hatching codes, but it was possible to display only four colors on the graphic screen, as the IBM color graphics adapter's medium resolution mode supports only four colors. The full seven colors provided by Symphony can be regenerated using a graphics plotter with a seven or more pen carousel. To obtain good hard copy plots of graphic images, the IBM-XT was interfaced to some extent with the existing HP-7721B plotters. The problem encountered here was that Symphony does not provide a driver for this plotter. The task to write the driver for the plotter was abandoned since no adequate guidelines were found in the Symphony manual on how to create new driver sets or how to save the Symphony screen images. All graphs included in this work were generated using Symphony and plotted using the HP-7475A plotters. These graphs demonstrate the effective use of a microcomputer for good quality graphics.

In the final stage of the work, the possibility of applying the package in a real time environment was considered. However, facilities to work with process computers in the control room of a real process plant were not available. In the university no sophisticated process simulator was available such that data could be extracted from the process, analysed using the process identification package and appropriate control action be fed back to the

process. However, an interface was established with the university DPC via a modem. Unfortunately there was no simulation package available which could work interactively with the present package. However this mainframe-to-microcomputer link adequately demonstrates the feasibility of linking the microcomputer to a process control computer.

Chapter 6

CONCLUDING REMARKS

6.1 Conclusions

The process identification package described here is a timely development for on-line data analysis and modeling. The plant diagnostic tool is implemented on a popular IBM Microcomputer which is fast and portable, and is free form system priorities and interrupts characteristic of process control and time shared systems. Now time series models can be determined on-line in the process control room and can be readily used for evaluation and analysis. In the present work all the necessary groundwork for implementing the package in real-time environment has been carried out.

Time series modeling techniques acted as stimulants for developing new hardware and software strategies to complement and enhance the power of analysis. The combination of interactive procedures of Box and Jenkins and the interactive capabilities of the microcomputer were combined to create a powerful but friendly package for time series analysis.

The Box and Jenkins technique was found to adapt readily to different classes of time series models. Both stationary and non-stationary data can be handled using the package.

Single input single output transfer function models generated in the analysis are parsimonious in nature. Lead time forecasts using the leading indicator have shown close agreement with the actual output. The important process characteristic of dead time, progress gain and impulse response are extracted by the technique and can be used for controller tuning and design.

Difficulties can arise in systems with large disturbances. Significant peaks from correlation plots for higher order mixed ARMA(n,m) models can be difficult to detect because of diverse patterns. Through the interactive framework of the package different possible model orders can be easily tried, but the whole process may be time consuming and require a high degree of user interaction.

Pandit and Wu technique, also implemented here offers an alternative modeling strategy. High order models and sets of data with high noise content were adequately modeled using this automatic approach. The state variable approach of the technique enabled in identification of the open or closed loop structure of the system. In view of its high computational requirements, this technique can be utilized as an off-line method. Multivariable modeling and fitting to specific model orders ca be conveniently carried out with the aid of this batch technique.

Transfer function modeling is the main function of package, but since univariate time series models belong to a subset of transfer function models, the modeling procedure of univariate time series an inherent part of the package.

Though emphasis was placed on preparing an on-line process analysis system, time series methods are widely applicable to economics, social sciences and planning, the developed package can be easily applied for socio-economic models.

Most engineering techniques require fast computation, analysis using visual displays, interface with the process and a need to interact with all the above facilities under a common interactive structure. The hardware/software system adopted on the present work can readily be adapted to cover other computations in science and engineering.

6.2 Future Work:

The present work has given insight to new areas of research, application and development where further work can be carried out.

The present package can be broadened to include transfer function modeling of general multivariable systems. Promising methods for simplifying the dauntingly large number of parameters in multiple time series have been attempted by

Granger and Newbold [17] and Tiao and Box [41]. A microcomputer based multivariable time series capability will greatly enhance package's sophistication and capabilities.

A wide variety of systems represented by time series models are assumed to be linear and a function of one variable time. However many engineering systems are non-linear. Development of non-linear identification algorithms for a restricted class of nonlinearities is an urgent need.

The non-linear estimation routines in time series modeling consume the largest amount of computation time and this is most significant on microcomputers. More efficient algorithms for minimizing the time required for non-linear least squares can greatly cut down the modeling time.

The memory and speed limitations found in the present machine can be removed to a large extent by implementing the package on a faster machine (e.g the IBM-AT). The greatly increased memory capacity, execution speed and very high speed of process interface of new generation machines offers the possibility of implementing time series methods in direct digital control.

The availability of multi-tasking environment managers or shells is a recent trend in system software. Shells support concurrent operation of multiple operations. Due to the

diverse tasks in the package, shells could thus provide strategies to perform transfer function analysis of different series at the same time.

User and system interface in the package is friendly and interactive. These distinctive features can allow the package to be easily implemented in CAI. It can be used as an instructional , demonstrative and research tool in time series methodology.

Access to process control rooms of local process plants will provide means for demonstrating the effectiveness and usefulness of the package. Routines for data smoothing, fitting of missing values and eliminating jumps [56,58] will be required when real time implementation is carried out.

Models obtained here can be made applicable in the areas of controller tuning using deterministic and stochastic (minimal variance) methods, special controllers such as cascade and feedforward, and multivariable control system design. Research is in progress to provide a comprehensive suite of routines on the microcomputer for this purpose [43,57].

REFERENCES

1. N.E. Gough, M.S. Ahmed and M.R. Al-Hafeez "Development of an on-line Microcomputer based Plant Diagnostic Tool", Proposal NO.22024, UPM Research Institute, Dhahran, 1984.

2. R.Nelson and P.Needle, "How much Computing Power do you need?", Personel Computing, pp.80-93, April 1985.

3. M.Kendall, "Time Series", London, Charles Griffin & Co.Ltd.,1984.

4. S.M.Pandit and S.M.Wu, "Time Series and System Analysis with Applications", New York, John Wiley & Sons, 1983.

5. D.P.Reilly, "Recent Experiences with an Automatic Box-Jenkins Modeling Algorithm", Proc.Int.Conf. on Time Series Analysis, Houston, August 1980.

6. G.E.P.Box and G.M.Jenkins, "Time Series Analysis Forecasting Control", San Francisco, Holden-Day 1976.

7. E.Greenberg and C.E. Webster Jr., "Advanced Ecnometric: A bridge to the literature", New York, John Wiley & Sons, 1983.

8. J.G.De-Gooijer and R.M.J. Hauts, "The Corner Method: An Investigation of an Order Discrimination Procedure for General ARMA Processes", J.Opr.Res.Soc., Vol.32, pp. 1039 - 1042, 1982.

9. W.L.Gray, G.Kelley and D.McIntire, "New Approach to ARMA Modeling", Comm. in Stat., B7, pp.1-77, 1978.

10. O.D.Anderson, "A New Approach to ARMA Modeling: Some Comments", Proc. Int.Conf.Analysing Time Series, Channel Islands, Oct.1979.

11. J.M.Beguin, C.Gourieroux and A.Monfort, "Identification of a Mixed Auto-Regressive Moving-Average Process; The Corner Method", Proc.Int.Conf. on Time Series, Nottingham, March-1979.

12. M.S.Phadke and S.M.Wu, "Identification of Multi-Input Multi-Output Transfer Function and Noise Model of a Blast Furnace from Closed Loop Data", IEEE Trans. on Automatic Control, Vol.10(6), pp.943-951, 1974.

13. L.M.Lui and D.M.Hanssens, "Identification of Multi Input Transfer Function Models", Comm. in Stat., A11(3), pp.297-314, 1982.

14. M.B.Priestley, "Estimation of Transfer Function in Closed Loop Stochastic Processes", Autometica, Vol.(5), pp.623-632, 1969.

15. I.D.Haugh and G.E.P. Box, "Identification of Dynamic Regression (Distributed Lag) Models Connecting Two Time Series", J.of Am.Stat. Asoc., Vol.72, pp.121-130, 1977.

16. A.Fask and P.B.Robinson, "Identification of Multivariate Econometric Models", Am.Stat. Asoc., Proc. of Buss.and Econ.Stat. Sect., pp.284-289, 1977.

17. C.W.J.Granger and P.New Bold, "Identification of Two Way Causal System", Frontiers in Quantitative Economics, Vol.IIIA, Amesterdam, North Holland, 1977.

18. S.M.Pandit and S.M. Wu, "Modeling and Analysis of Closed Loop Systems from Operating Data", Technometrics, Vol.19-4, pp.477-485, 1977.

19. D.C.Montgomery and L.A.Johnson, "Forecasting and Time Series Analysis", New York, McGraw-Hill Book Co., 1976.

20. R.McCleary and R.A.Hay Jr., "Applied Time Series Analysis", London, Sage Publications, 1980.

21. "The IMSL Libraries", IMSL, Houston, 1984.

22. H.J.Newton, "What should your Time Series Analysis Program Do?", Comp.Sc.and Stat. 15th Interface, North Holland Publishing Co., 1983.

23. H.Akaike, Information Theory and Extension of the Maximum Likelihood Principle, 2nd Int.Symp. on Inf.Theory, Budapest: Akademiai, Kiado, 267-281, 1973.

24. B.F.Ryan and T.A.Ryan Jr., "Minitab or Microcomputers", Comp. Sc. and Stat: 13th Interface, Springer Verlag, New

York, 83-85, 1981.

25. S.Kaltio, "A System for Time Series Analysis", Int.Conf. on Time Series, Nottingham, March 1979.

26. J.Prins,"Interactive Multivariate Time Series Analysis", Int. Conf. on Analysing Time Series, Channel Islands, Oct.1979.

27. N.W.Polhemms, "Analysis and Modeling of Time Series via Interactive Computer Graphics, Int.Conf. on Analysing Time Series, Channel Islands, Oct.1979.

28. G.M.Jenkins, "Modeling and Forecasting Time Series", Channel Islands, Gwilym Jenkins & Partners Ltd., 1979.

29. J.C.Nash, "Compact Numerical Methods for Computers: Linear Algebra and Function Minimization", Bristol, Adam Hilger Ltd., 1979.

30. B.L.Boxerman, "Time Series and Forecasting", North Scituate,Mass. Duxbury Press, 1979.

31. S.M.Wu, "Dynamic Data Systems - A New Modeling Technique", Trans. ASME, Series B, Vol.99, No.3, pp 708-714.

32. S.M.Pandit, "Data Dependent Systems and Experimental Smoothing", Int.Conf. on Analysing Time Series, Channel Islands, Oct.1979.

33. S.M.Pandit, "Data Dependent Systems, Modeling and Optimal Control via Time Series", Ph.D.Thesis, Univ. of Wisconsin, Madison, Wisconson, 1973.

34. "IBM Personal Computer Math Co-Processor Option (1 501002)", IBM Corp., June 1983.

35. K.Koessel, "An Introduction to RAM Disks", pp.59-61, PC-World, Feb.1985.

36. D.Holzman, "Using MS Fortran on the PC", PC Magazine, Aug.21,1984.

37. E.Barans, "Symphony Framework Face off", PC-Magazine, pp.128-135, Aug.14,1984.

38. R.A.Adey, "Menu Input Generating System for the FORTRAN Program", Eng.Sofware III, Springer Verlag, New York, 1983.

39. "Respond Communication Software", PC-World, p-98, Feb.1985.

40. J.C.Nash, "Where to find the Nuts & Bolts:Sources of Software", Comp.Sc. and Stat:14 th Interface North Holland Pub.Co. 1982.

41. C.Tiao, and G.E.P.Box, "Modeling Multiple Time Series with Applications", J.Am.Stat.Assoc.Vol(76), pp.802-816, 1981.

42. C.Warren, "Environmental Software Opens System Windows", MinMicro System, Mar.1985.

43. N.E. Gough, "Computer Aided Design of Control Systems using Microsoft Basic", Proc.ACI'83, Copenhagen, 1983.

44. R.Skrokov, "Mini and Micro Computer Control in Industrial Processes" , Van Nostrand Reinhold Comp., New York, 1980.

45. W.H.Rey, "Multivariable Process Control - A Survey", Computers and Chemical Engineering, Vol.7(4), pp.1367-1394.

46. C.R.Cutler, and R.T.Perry, "Real Time Optimization with Multivariable Control is Required to Maximize Profits", Computer and Chemical Engineering, Vol.7, No.5, pp.663-667, 1983.

47. M.S. Ahmed, "Computationally Efficient IV algorithm for Structure and Parameter Identification of Linear System" Proc. IEE, Part D, March 1985.

48. C.R.Nelson, "Applied Time Series Analysis for Managerial Forecasting", Holden day, San Francisco, 1973.

49. J.H.Beguin, C.Gouvieroux and A.Monfort, "The Applicability of the Corner Method: A reply.J.Opr.Res.Soc.Vol.32, pp.1042-1045.

50. D.W.Marquardt, "An Algorithm for Least Squares Estimation

of Non Linear Parameters", J.Soc.Ind.Appl.Math II, 431-441.

51. R.Fletcher, "A Modified Marquardt Subroutine for Nonlinear Least Squares", U.K.Atomic Energy Research Establishment, Harwell, Report R6799, 1970.

52. J.C.Nash, "Nonlinear Estimation Using a Microcomputer", Comp.Sc.and Stat: 13th Interface, Springer Verlag, New York, 363-366.

53. S.Shahhabuddin, "Time Series - An Interactive Computer Program for Time Series Analysis", Am.Stat.,Vol.33(4), p.223, Nov.1979.

54. P.Newbold, "Time-Series Model building and Forecasting: A Survey", TIMS Studies in Management Sciences, 12, pp.59-73, 1979.

55. C.M.Stralkowski, R.E.DeVor and S.M.Wu, "Charts for the Interpretation and Estimation of the Second Order Moving Average and Mixed First Order Auto Regressive Moving Average Models", Technometrics, Vol.16(2), pp.275-285, 1974.

56. T.Vo-Dai, "Time Series Analysis with missing or aberrant data", Proc.Comp.Stat, 4th Sym.Edinburgh, 1980.

57. N.E.Gough and R.S. Al-Thiga, "Small-Computer Procedures

for training control engineers", 5th Int.Symp."Computer at University", Cavtat,1983.

58. P.M.Robinson, "Estimation and Forecasting for Time Series Containing Censored or Missing Observations", Proc.Int.Conf.Time Series, March 1979.

59. J.D.Henstridge, "TSA, A package for time series analysis and manipulation", Proc.Comp.Stat., 4th Symp., Edinburgh, 1980.

60. T.Pukkila and S.F.Tampene, "Time Series Analytical Operations of SURVO/71[", Proc.Comp.Stat., 4th Symp., Edinburgh, 1980.

61. M.R.Al-Hafeez, "Process Identification Package: A User's Guide & Report", Department of Systems Engineering,Internal Report, UPM, Dhahran, 1985.

APPENDIX

Appendix A

NON LINEAR ESTIMATION ALGORITHM

Purpose: To minimize a sum of squares of m non-linear functions r_i of n variables, that is to

$$\text{minimize } \sum_{i=1}^{m} [r_i(x_1, x_2, \ldots, x_n)]^2.$$

The algorithm described is the Hartley's modification to the Gauss-Newton method.

Algorithm:

Step 1: Calculate the functions r_i's, i=1,2,...,m given the initial values of x(t)'s where l=iteration number equal to 1.

Step 2: Calculate the numerical derivative by

$$\partial r/\partial x \simeq [r(x+h) - r(x)]/h$$

$$H_k^T = [\partial r_k/\partial x_1, \ldots, \partial r_k/\partial x_n]$$

$$A_{ij} = \sum_{k=1}^{m} H_k H_k^T \quad \text{for } j=1,2,\ldots,i;\ i=1,2,\ldots,n$$

and

$$V_i = \sum_{k=1}^{m} r_k H_k \qquad \text{for } i=1,2,\ldots,n$$

Step 3: $x(l+1) = x(l) + \alpha A^{-1} V$

Take α as <1 preferably between 0.05 and 0.5, to ensure convergence.

Step 4: If

$$\left[\sum_{k=1}^{m} r_k^2(l+1) - \sum_{k=1}^{m} r_k^2(l)\right] / \sum_{k=1}^{m} r_k^2(l) < 0.01$$

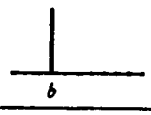
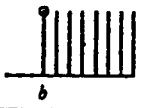
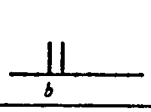
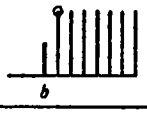
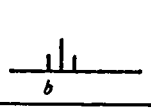
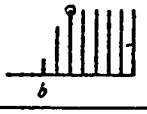
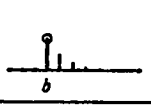
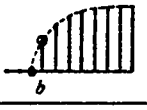
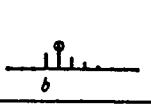
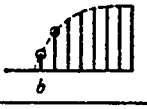
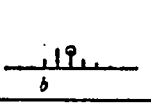
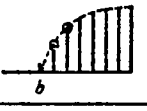
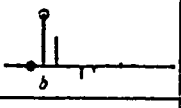
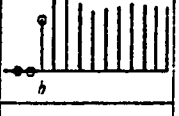
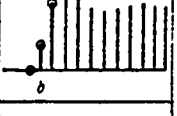
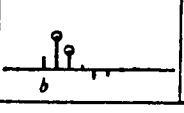
then stop.

Else

$l=l+1$ go to Step 2.

APPENDIX - B

IMPULSE & STEP RESPONSE PATTERNS

r, s, b	∇ Form	B Form	Impulse Response v_j	Step Response $V_i = \sum_{j=0}^{i} v_j$
003	$Y_t = X_{t-3}$	$Y_t = B^3 X_t$		
013	$Y_t = (1 - .5\nabla) X_{t-3}$	$Y_t = (.5 + .5B) B^3 X_t$		
023	$Y_t = (1 - \nabla + .25\nabla^2) X_{t-3}$	$Y_t = (.25 + .50B + .25B^2) B^3 X_t$		
103	$(1 + \nabla) Y_t = X_{t-3}$	$(1 - .5B) Y_t = .5B^3 X_t$		
113	$(1 + \nabla) Y_t = (1 - .5\nabla) X_{t-3}$	$(1 - .5B) Y_t = (.25 + .25B) B^3 X_t$		
123	$(1 + \nabla) Y_t = (1 - \nabla + .25\nabla^2) X_{t-3}$	$(1 - .5B) Y_t = (.125 + .25B + .125B^2) B^3 X_t$		
203	$(1 - .25\nabla + .5\nabla^2) Y_t = X_{t-3}$	$(1 - .6B + .4B^2) Y_t = .8B^3 X_t$		
213	$(1 - .25\nabla + .5\nabla^2) Y_t = (1 - .5\nabla) X_{t-3}$	$(1 - .6B + .4B^2) Y_t = (.4 + .4B) B^3 X$		
223	$(1 - .25\nabla + .5\nabla^2) Y_t = (1 - \nabla + .25\nabla^2) X_{t-3}$	$(1 - .6B + .4B^2) Y_t = (.2 + .4B + .2B^2) B^3 X_t$		

Examples of impulse and step response functions with gain $g = 1$

APPENDIX - C

GAS FURNACE DATA.

t	INPUT	OUTPUT	t	INPUT	OUTPUT
1	-0.109	53.8	51	1.608	46.9
2	0.000	53.6	52	1.905	47.8
3	0.178	53.5	53	2.023	48.2
4	0.339	53.5	54	1.815	48.3
5	0.373	53.4	55	0.535	47.9
6	0.441	53.1	56	0.122	47.2
7	0.461	52.7	57	0.009	47.2
8	0.348	52.4	58	0.164	48.1
9	0.127	52.2	59	0.671	49.4
10	-0.180	52.0	60	1.019	50.6
11	-0.588	52.0	61	1.146	51.5
12	-1.055	52.4	62	1.155	51.6
13	-1.421	53.0	63	1.112	51.2
14	-1.520	54.0	64	1.121	50.5
15	-1.302	54.9	65	1.223	50.1
16	-0.814	56.0	66	1.257	49.8
17	-0.475	56.8	67	1.157	49.6
18	-0.193	56.8	68	0.913	49.4
19	0.088	56.4	69	0.620	49.3
20	0.435	55.7	70	0.255	49.2
21	0.771	55.0	71	-0.280	49.3
22	0.966	54.3	72	-1.080	49.7
23	0.875	53.2	73	-1.551	50.3
24	0.891	52.3	74	-1.799	51.3
25	0.987	51.6	75	-1.825	52.9
26	1.263	51.2	76	-1.456	54.4
27	1.775	50.8	77	-0.944	56.0
28	1.976	50.5	78	-0.570	56.9
29	1.934	50.0	79	-0.431	57.5
30	1.866	49.2	80	-0.577	57.3
31	1.872	48.4	81	-0.960	56.6
32	1.767	47.9	82	-1.616	56.0
33	1.608	47.6	83	-1.975	55.4
34	1.265	47.5	84	-1.891	55.4
35	0.790	47.5	85	-1.746	56.4
36	0.360	47.6	86	-1.474	57.2
37	0.115	48.1	87	-1.201	58.0
38	0.088	49.0	88	-0.927	58.4
39	0.331	50.0	89	-0.524	58.4
40	0.645	51.1	90	0.040	58.1
41	0.960	51.8	91	0.738	57.7
42	1.409	51.9	92	0.943	57.0
43	2.670	51.7	93	0.930	56.0
44	2.834	51.2	94	1.006	54.7
45	2.812	50.0	95	1.137	53.2
46	2.483	48.3	96	1.198	52.1
47	1.929	47.0	97	1.054	51.6
48	1.485	45.8	98	0.595	51.0
49	1.214	45.6	99	-0.080	50.5
50	1.239	46.0	100	-0.314	50.4

t	INPUT	OUTPUT	t	INPUT	OUTPUT
101	-0.288	51.0	151	-0.049	53.2
102	-0.153	51.8	152	0.060	53.9
103	-0.109	52.4	153	0.161	54.1
104	-0.187	53.0	154	0.301	54.0
105	-0.255	53.4	155	0.517	53.6
106	-0.227	53.6	156	0.566	53.2
107	-0.007	53.7	157	0.560	53.0
108	0.254	53.8	158	0.573	52.8
109	0.330	53.8	159	0.592	52.3
110	0.102	53.8	160	0.671	51.9
111	-0.423	53.3	161	0.933	51.6
112	-1.139	53.0	162	1.337	51.6
113	-2.275	52.9	163	1.460	51.4
114	-2.594	53.4	164	1.353	51.2
115	-2.716	54.5	165	0.772	50.7
116	-2.510	56.4	166	0.218	50.0
117	-1.790	58.0	167	-0.237	49.4
118	-1.346	59.4	168	-0.714	49.3
119	-1.081	60.2	169	-1.099	49.7
120	-0.910	60.0	170	-1.269	50.6
121	-0.876	59.4	171	-1.175	51.8
122	-0.885	58.4	172	-0.676	53.0
123	-0.800	57.6	173	0.033	54.0
124	-0.544	56.9	174	0.556	55.3
125	-0.416	56.4	175	0.643	55.9
126	-0.271	56.0	176	0.484	55.9
127	0.000	55.7	177	0.109	54.6
128	0.403	55.3	178	-0.310	53.5
129	0.841	55.0	179	-0.697	52.4
130	1.285	54.4	180	-1.047	52.1
131	1.607	53.7	181	-1.218	52.3
132	1.746	52.8	182	-1.183	53.0
133	1.683	51.6	183	-0.873	53.8
134	1.465	50.6	184	-0.336	54.6
135	0.993	49.4	185	0.063	55.4
136	0.648	48.9	186	0.084	55.9
137	0.577	48.5	187	0.000	55.9
138	0.577	48.7	188	0.001	55.2
139	0.632	49.2	189	0.209	54.4
140	0.747	49.8	190	0.536	53.7
141	0.900	50.4	191	0.782	53.6
142	0.993	50.7	192	0.858	53.6
143	0.968	50.9	193	0.918	53.2
144	0.790	50.7	194	0.862	52.5
145	0.399	50.5	195	0.416	52.0
146	-0.161	50.4	196	-0.336	51.4
147	-0.553	50.2	197	-0.959	51.0
148	-0.603	50.4	198	-1.813	50.9
149	-0.424	51.2	199	-2.378	52.4
150	-0.194	52.3	200	-2.499	53.5

t	INPUT	OUTPUT	t	INPUT	OUTPUT
201	-2.473	55.6	251	0.185	56.3
202	-2.330	58.0	252	0.662	56.4
203	-2.053	59.5	253	0.709	56.4
204	-1.739	60.0	254	0.605	56.0
205	-1.261	60.4	255	0.501	55.2
206	-0.569	60.5	256	0.603	54.0
207	-0.137	60.2	257	0.943	53.0
208	-0.024	59.7	258	1.223	52.0
209	-0.050	59.0	259	1.249	51.6
210	-0.135	57.6	260	0.824	51.6
211	-0.276	56.4	261	0.102	51.1
212	-0.534	55.2	262	0.025	50.4
213	-0.871	54.5	263	0.382	50.0
214	-1.243	54.1	264	0.922	50.0
215	-1.439	54.1	265	1.032	52.0
216	-1.422	54.4	266	0.866	54.0
217	-1.175	55.5	267	0.527	55.1
218	-0.813	56.2	268	0.093	54.5
219	-0.634	57.0	269	-0.458	52.8
220	-0.582	57.3	270	-0.748	51.4
221	-0.625	57.4	271	-0.947	50.8
222	-0.713	57.0	272	-1.029	51.2
223	-0.848	56.4	273	-0.928	52.0
224	-1.039	55.9	274	-0.645	52.8
225	-1.346	55.5	275	-0.424	53.8
226	-1.628	55.3	276	-0.276	54.5
227	-1.619	55.2	277	-0.158	54.9
228	-1.149	55.4	278	-0.033	54.9
229	-0.488	56.0	279	0.102	54.8
230	-0.160	56.5	280	0.251	54.4
231	-0.007	57.1	281	0.280	53.7
232	-0.092	57.3	282	0.000	53.3
233	-0.620	56.8	283	-0.493	52.8
234	-1.086	55.6	284	-0.759	52.6
235	-1.525	55.0	285	-0.824	52.6
236	-1.858	54.1	286	-0.740	53.0
237	-2.029	54.3	287	-0.528	54.3
238	-2.024	55.3	288	-0.204	56.0
239	-1.961	56.4	289	0.034	57.0
240	-1.952	57.2	290	0.204	58.0
241	-1.794	57.8	291	0.253	58.6
242	-1.302	58.3	292	0.195	58.5
243	-1.030	58.6	293	0.131	58.3
244	-0.916	58.8	294	0.017	57.8
245	-0.798	58.8	295	-0.192	57.3
246	-0.867	58.6	296	-0.262	57.0
247	-1.047	58.0			
248	-1.123	57.4			
249	-0.876	57.0			
250	-0.395	56.4			

APPENDIX - D

DISTILLATION COLUMN DATA.

t	TS#1	TS#2	TS#3	t	TS#1	TS#2	TS#3
1	182.6	83.6	31.7	51	182.1	62.1	27.2
2	182.7	83.7	31.3	52	182.1	63.5	26.3
3	182.8	83.9	31.8	53	181.9	64.9	26.3
4	183.0	82.7	30.3	54	181.8	65.4	26.0
5	182.9	81.1	31.4	55	181.7	66.6	26.3
6	182.5	81.5	32.1	56	181.8	66.8	25.4
7	182.3	80.6	32.2	57	181.8	67.3	25.3
8	182.1	80.8	32.1	58	181.9	67.0	26.1
9	182.1	81.9	32.0	59	182.0	66.8	26.3
10	182.1	81.9	31.2	60	182.1	66.3	25.9
11	182.4	80.6	30.6	61	182.3	65.3	25.6
12	182.6	80.3	31.4	62	182.7	63.4	24.7
13	182.8	80.9	31.2	63	183.4	57.8	22.1
14	182.9	79.9	30.7	64	183.5	54.3	24.3
15	183.1	79.5	28.4	65	182.9	48.2	27.9
16	183.5	74.4	26.3	66	182.0	45.9	29.3
17	183.6	73.0	27.5	67	181.4	47.7	27.5
18	182.4	72.6	27.2	68	180.3	53.9	30.4
19	180.9	71.2	26.7	69	180.2	58.8	26.1
20	180.0	70.4	27.6	70	180.9	60.1	22.9
21	179.5	71.1	29.8	71	181.5	59.8	22.8
22	179.7	72.0	28.7	72	181.7	60.2	24.8
23	180.2	70.9	27.4	73	181.8	62.4	27.0
24	180.8	70.8	28.4	74	182.2	63.8	25.5
25	181.3	72.1	29.0	75	182.6	61.3	24.8
26	181.7	73.4	29.0	76	183.3	59.7	24.0
27	181.6	74.8	28.5	77	183.8	56.9	23.0
28	181.8	73.3	27.8	78	183.4	53.1	27.9
29	182.3	72.9	27.6	79	182.3	56.0	30.8
30	183.0	71.0	28.4	80	181.7	59.7	30.1
31	183.2	72.2	31.9	81	181.8	61.3	25.0
32	183.1	75.2	31.6	82	181.7	63.7	25.3
33	182.9	75.4	28.8	83	182.0	62.7	25.4
34	182.8	73.9	28.8	84	182.9	59.9	24.7
35	182.7	73.4	29.0	85	184.1	59.6	21.8
36	182.6	73.9	27.4	86	183.8	59.5	23.3
37	182.6	71.2	25.3	87	182.8	59.6	30.0
38	182.7	67.7	25.2	88	181.8	59.7	28.1
39	182.2	65.8	26.1	89	181.8	59.2	26.1
40	181.5	63.9	28.4	90	181.7	59.5	27.0
41	181.2	62.7	29.3	91	181.7	62.3	27.4
42	181.2	62.1	29.2	92	181.6	63.6	27.3
43	181.2	63.1	28.5	93	181.4	65.6	29.0
44	181.3	62.6	27.3	94	181.3	70.4	29.8
45	181.3	62.5	26.9	95	181.5	72.3	29.7
46	181.2	63.3	27.1	96	181.8	73.7	30.9
47	181.3	64.7	25.9	97	181.8	75.6	32.5
48	181.5	64.2	24.9	98	181.4	77.7	32.9
49	181.7	63.2	25.0	99	181.1	79.7	31.7
50	182.0	61.4	25.7	100	181.4	81.0	30.6

t	TS#1	TS#2	TS#3	t	TS#1	TS#2	TS#3
101	181.8	82.8	31.4	151	181.9	76.7	29.8
102	182.0	83.0	30.9	152	182.0	75.5	30.8
103	182.4	83.6	30.0	153	182.0	74.2	30.7
104	182.5	84.0	30.5	154	181.6	74.2	32.3
105	182.7	83.2	29.3	155	181.3	75.5	31.3
106	182.6	81.8	29.8	156	181.1	76.2	31.0
107	182.7	79.4	27.9	157	181.2	77.3	30.9
108	182.5	76.7	27.8	158	181.4	77.9	30.2
109	182.2	72.0	27.8	159	182.2	78.1	28.1
110	181.5	67.8	32.0	160	183.9	72.2	22.5
111	180.6	66.0	34.3	161	182.7	69.1	19.9
112	180.0	67.9	33.4	162	179.8	66.1	20.6
113	179.6	70.8	32.8	163	177.7	62.7	23.0
114	179.8	75.5	32.2	164	177.2	59.6	29.5
115	180.1	77.6	30.9	165	177.8	59.3	33.2
116	180.7	79.2	29.2	166	178.7	64.2	32.1
117	180.5	79.0	27.6	167	179.8	68.9	30.5
118	182.2	77.7	27.0	168	180.7	72.1	29.2
119	182.6	77.6	28.2	169	181.3	72.1	29.6
120	182.8	77.3	29.7	170	181.9	75.3	31.1
121	182.7	77.3	29.9	171	182.5	77.0	29.1
122	182.5	75.3	30.2	172	183.0	75.6	28.1
123	182.4	75.1	29.4	173	183.1	76.4	30.0
124	182.2	74.0	29.7	174	182.9	77.2	30.6
125	182.1	73.8	30.3	175	182.6	76.1	31.4
126	181.7	74.8	30.6	176	182.3	76.6	32.2
127	181.6	74.1	29.5	177	182.1	77.0	31.9
128	181.5	74.1	29.5	178	181.9	77.8	32.5
129	181.4	74.4	29.8	179	181.7	79.3	31.9
130	181.3	75.3	28.9	180	181.7	79.0	30.9
131	181.2	75.5	28.8	181	181.6	77.9	31.3
132	181.2	74.8	28.6	182	181.6	79.8	31.7
133	181.2	75.4	29.0	183	181.6	78.7	31.3
134	181.2	75.6	29.9	184	181.6	78.9	31.1
135	181.0	78.1	30.5	185	181.7	78.7	30.7
136	181.1	76.9	29.1	186	181.7	78.3	30.6
137	181.4	76.8	28.9	187	181.7	77.9	30.5
138	181.6	76.8	29.9	188	181.7	76.4	30.3
139	181.5	79.0	31.3	189	181.6	76.5	30.7
140	181.3	78.9	30.4	190	181.5	78.1	31.5
141	181.5	77.3	28.5	191	181.3	78.4	31.6
142	181.6	77.9	30.2	192	181.2	75.9	30.8
143	181.4	78.9	31.8	193	181.2	76.2	30.5
144	181.1	78.8	31.1	194	181.4	75.5	30.4
145	181.1	78.8	29.5	195	181.5	75.7	30.6
146	181.4	78.2	29.8	196	181.5	76.0	31.6
147	181.7	77.4	30.9	197	181.6	77.0	31.0
148	181.7	78.4	30.7	198	181.7	77.5	30.8
149	181.7	78.1	30.0	199	181.8	78.4	31.0
150	181.7	76.6	29.9	200	181.9	78.7	30.9

t	TS#1	TS#2	TS#3
201	182.0	79.3	31.1
202	182.1	80.9	31.1
203	182.1	80.9	31.3
204	182.2	82.0	31.1
205	182.3	81.6	30.7
206	182.4	80.8	30.4
207	182.6	80.3	30.4
208	182.5	79.5	31.2
209	182.4	79.8	31.7
210	182.1	78.8	31.4
211	181.9	77.9	31.9
212	181.7	78.0	32.5
213	181.6	78.9	32.5
214	181.5	78.8	31.5
215	181.6	79.2	31.0
216	181.8	78.8	32.2
217	181.9	79.8	32.7
218	182.0	79.5	33.0
219	182.0	80.7	32.9
220	182.1	81.5	32.8
221	182.4	81.7	32.0
222	182.6	81.7	33.0
223	182.7	82.5	33.0
224	182.7	82.5	33.0
225	182.7	82.3	33.8
226	182.9	81.6	32.8
227	182.7	80.4	32.1
228	182.7	78.5	33.3
229	183.3	78.5	33.0
230	183.7	79.8	32.2
231	183.8	80.6	32.2
232	184.0	81.5	31.1
233	184.5	81.7	30.4
234	184.7	82.0	29.4
235	183.3	81.1	30.1
236	187.9	80.1	31.8
237	183.7	79.8	32.1
238	183.2	80.7	31.8
239	183.0	80.5	31.5
240	182.8	81.1	30.9
241	182.7	81.6	30.6
242	182.8	79.7	30.3
243	183.1	77.7	29.8
244	183.4	75.9	29.5
245	183.3	72.7	28.6
246	183.3	70.6	28.1
247	183.0	67.8	29.2
248	182.6	66.1	30.2
249	182.3	67.7	30.5
250	182.2	68.0	30.3
251	182.1	69.7	30.6
252	182.1	72.4	29.9
253	182.3	73.4	28.6
254	182.7	74.0	28.8
255	183.0	74.4	28.5
256	183.3	73.5	26.7
257	183.7	72.1	26.3
258	183.8	72.0	28.0
259	183.5	71.1	28.8
260	183.2	69.8	28.1
261	182.9	68.9	27.8
262	182.9	69.2	28.3
263	182.7	68.7	28.7
264	182.7	67.5	28.5
265	182.6	68.2	28.4
266	182.5	67.5	27.9
267	182.4	67.2	28.4
268	182.5	68.1	28.5
269	182.6	66.1	28.6
270	182.7	66.9	28.9
271	182.9	67.4	28.0
272	183.0	67.0	28.3
273	183.1	68.2	28.3
274	183.1	67.7	26.3
275	183.2	67.9	28.6
276	183.1	69.4	29.0
277	183.1	69.6	29.5
278	183.1	69.2	27.2
279	183.2	67.3	27.8
280	183.1	67.6	28.0
281	182.9	65.9	28.4
282	182.6	66.8	26.7
283	182.5	66.9	28.2
284	182.5	66.1	28.2
285	182.4	67.2	27.7
286	182.3	66.9	28.1
287	182.4	66.6	28.6
288	182.6	68.5	29.1